FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Aplicação móvel para a prestação de serviços *peer-to-peer* no âmbito da *share economy*

Cristiano Filipe Teixeira Alves

Mestrado Integrado em Engenharia Informática e Computação

Orientador: António Miguel Pontes Pimenta Monteiro

Junho de 2015

Aplicação móvel para a prestação de serviços *peer-to-peer* no âmbito da *share economy*

Cristiano Filipe Teixeira Alves

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: José Manuel De Magalhães Cruz

Vogal Externo: Helena Rodrigues

Orientador: António Miguel Pontes Pimenta Monteiro

Junho de 2015

Resumo

As *share economies* começaram a ter uma maior dimensão e impacto no final da primeira década do século XXI, aproveitando o despertar das novas tecnologias e da Internet, que nos mantêm cada vez mais interligados, para recriar um modelo de negócio já existente.

Esse modelo de negócio permitia que qualquer pessoa prestasse serviços ou vendesse/alugasse bens, que estavam subutilizados e, com isso, criar rendimento. O que mudou neste modelo de negócio foi a sua escala passando de uma escala muito reduzida para uma escala global podendo agora ser potenciado de modo a que mais pessoas possam beneficiar destes serviços. Com o aumento da escala das *share economies* criou-se um novo tipo de mercado, o mercado *peer-to-peer*. Este mercado funciona, na maioria dos casos, *online* e permite que pessoas possam encontrar serviços prestados por outras pessoas. Esses mercados têm como ponto principal a confiança, sendo fundamental que existam mecanismos de criação dessa confiança.

O objetivo desta dissertação será analisar este modelo de negócio em especial a prestação de serviços *peer-to-peer*, de modo a criar uma aplicação de prova de conceito, que suportará, como exemplo, estafetas de entrega de refeições. Será necessário explorar todos os problemas relacionados com o modelo de negócio em questão, sendo o grande desafio desta área de negócio a criação de confiança por parte dos utilizadores. Assim sendo, será fundamental investigar e conceber formas de aumentar a confiança dos utilizadores nos prestadores de serviços e no próprio sistema.

Outra área importante nas aplicações *share economy*, que também requer muita confiança por parte dos utilizadores, é a forma de pagamento do serviço. Para facilitar estas transações, muitas vezes os pagamentos não são em dinheiro mas sim utilizando a própria aplicação. Será então necessário garantir que todo o fluxo de pagamento decorre de forma correta e segura.

É também importante prever conflitos entre os vários utilizadores e dotar o sistema de regras que os permitam avaliar e facilmente solucionar. Nessas regras poder-se-ão utilizar vários sensores dos *smartphones*, como, por exemplo, o GPS.

Abstract

Share economies began to have a greater scale and impact at the end of the first decade of the 21st century taking advantage of the rising of the new technologies and the Internet, which keeps us increasingly interconnected. This allowed to recreate an already existing business model.

This business model allows any person to provide services or sell/rent goods that are not used and create income. The change in this business model is the scale. Now, it works on global scale and more people can benefit from these new services. This increased scale of share economies has created a new type of market, the peer-to-peer market. This market works, in most cases, online and allows people to find services provided by others. These markets have as main and fundamental point the trust between users. The existence of mechanisms for the establishment of such trust is though very important.

The goal of this dissertation will be reviewing this business model and in particular the provision of peer-to-peer services, so as to create a proof-of-concept application, to be used for courier delivery of meals. It will need to explore all the problems related to the business model in question, being the great challenge of this business area, and also creating trust on the part of users. Therefore, it is essential to investigate and devise ways to increase the trust of users in service providers and in the system itself.

Another important area in applications of share economy, is the form of payment of the service. To facilitate these transactions, often payments are not in cash but intermediated by the application itself. It will be necessary to ensure that all the payment stream is correctly and safely done.

It is also important to predict conflicts between multiple users and provide the system of rules that assess and easily solve them. These rules may be using various sensors of smartphones, such as the GPS.

Agradecimentos

No culminar desta dissertação, não posso deixar de agradecer a todas as pessoas que me ajudaram neste meu percurso académico.

Em primeiro lugar aos meus pais e a minha irmã, agradeço pelo apoio e pela paciência que tiveram ao longo destes anos, sem vocês não teria chegado até aqui.

Agradeço, também, à minha namorada, Teresa Vaz, por compreender a falta de tempo para lhe dar atenção, por me ouvir a falar das dificuldades que ia encontrando nesta caminhada e pelo apoio incondicional.

A todos os meus amigos, aos de sempre e aos novos que conheci neste percurso. Em especial ao Filipe Sousa, pelas dores de cabeça que tivemos juntos nos trabalhos que desenvolvemos.

Agraço, claro, ao meu orientador Professor Miguel Pimenta Monteiro. Pelo acompanhamento prestado e pela disponibilidade que sempre apresentou, respondendo a todas as dúvidas com a atenção devida.

Não posso deixar de agradecer à empresa que me acolheu no desenvolvimento desta dissertação – *Glazed Solutions*. Mas mais que agradecer à empresa devo agradecer às pessoas, nomeadamente aos Engenheiros Pedro Campos e Diogo Nunes, pelo apoio e acompanhamento prestado durante a realização da dissertação.

Agradeço a todas as pessoas que aceitaram participar nos testes de usabilidade da aplicação. O seu contributo foi muito importante para avaliar a aplicação desenvolvida.

Devo, também, o meu agradecimento a todos os professores que encontrei tanto na Faculdade de Engenharia como no meu percurso anterior a esta. Dos professores gostava de salientar o Professo António Augusto Sousa, antigo diretor do MIEIC, e o Professor Miguel Ângelo, professor no Colégio Internato dos Carvalhos. O primeiro pela preocupação com todos os estudantes, pondo sempre em primeiro o interesse deles. O segundo pelo que me ensinou durante o ensino secundário e por me ter incentivado a entrar na Faculdade de Engenharia.

Por fim agradeço a todas as outras pessoas que não mencionei mas que se cruzaram comigo neste percurso, que de uma maneira ou de outra fizeram que este percurso fosse especial.

A todos, um muito obrigado!

Cristiano Alves

“Machines take me by surprise with great frequency.”

Alan Mathison Turing

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

ACL	Access Control Lists
ANTRAL	Associação Nacional dos Transportadores Rodoviários em Automóveis Ligeiros
B2B	Business-to-business
B2C	Business-to-consumer
BAAS	Backend as a Service
C2B	Consumer-to-business
C2C	Consumer-to-consumer
CLP	Class-Level Permissions
CVV	Card Verification Value
EUA	Estados Unidos da América
FAQ	Frequently Asked Questions
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
OPL	Open Programming Language
OS	Operation System
P2P	Peer-to-peer ou Person-to-person
PCI DSS	Payment Card Industry Data Security Standard
PIN	Personal identification number
REST	Representational State Transfer
SDK	Software Development Kit
SEPA	Single Euro Payments Area
SMS	Short Message Service
WWW	Word Wide Web

Capítulo 1

Introdução

No final da primeira década do século XXI, o mundo esteve mergulhado numa crise económica e, muitas empresas não resistiram a essa crise, acabando por encerrar, enquanto outras tiveram de reduzir o número de funcionários arrastando milhões de pessoas para o desemprego. Por outro lado, nos últimos anos, desde o final da década de 90, a humanidade assistiu a um grande desenvolvimento tecnológico e, ao mesmo tempo, esse desenvolvimento tornou a tecnologia mais acessível à população. Tornando-se normal em qualquer habitação existirem vários computadores com acesso à Internet. Mais tarde surgiram os *smartphones* que acrescentaram mais mobilidade à tecnologia.

Os dois fatores anteriores foram fundamentais no aparecimento da *share economy*, como a conhecemos hoje. A *share economy* permite partilhar bens ou serviços subutilizados com quem precisa deles, obtendo rendimentos com isso. Com a crise económica, esta forma de obter rendimentos era ideal para ser utilizada por desempregados que passavam por dificuldades e foi isso que algumas empresas reconheceram. Utilizando a tecnologia, criaram plataformas *online* que facilitam a utilização deste modelo de negócio e, por outro lado, aumentando-lhe a escala, passaram de um modelo negócio que era só aplicável a um nível local para um modelo de negócio global.

1.1 Enquadramento

Esta dissertação enquadra-se no domínio da *share economy*. Este modelo de negócio não é recente, porém, nos últimos anos foram-lhe aplicadas algumas alterações. Inicialmente, *share economies* não era mais do que a prestação de alguns serviços a familiares ou vizinhos, de forma a obter algum rendimento. Atualmente é um modelo de negócio que escalou, recorrendo às novas tecnologias, e tornando-se num modelo de negócio à escala global. No entanto, esta dissertação também se enquadra no domínio dos dispositivos móveis. Tendo como exemplo serviços *share economy* já no mercado, é fundamental o serviço apresentar a possibilidade de ser utilizado por meio de uma aplicação móvel. Estes dispositivos, além de estarem sempre junto das pessoas e serem mais acessíveis, apresentam também funcionalidades que em algumas aplicações desta área

são importantes, como por exemplo, os serviços de localização (GPS), que outros dispositivos, normalmente, não possuem.

1.2 Problema

Os serviços *share economy* já estão presentes em vários países e cobrem algumas áreas de atividade da sociedade. Porém, ainda existem setores onde estes serviços são poucos explorados, como é o caso do setor das entregas. Neste setor existem alguns serviços *share economy* porém, na maioria dos casos, não são exploradas todas as capacidades que este modelo de negócio permite. Existem muitos serviços que reduzem a comunicação entre o cliente e o prestador de serviço, perdendo assim a ideia de partilha que está presente na génese da *share economy*.

Embora existam já um grande número de aplicações seguindo este modelo de negócio, a maior parte está sediada nos EUA, e poucas se encontram sediadas na Europa. Estes serviços apresentam, em alguns casos, ambiguidades legais levando que, em alguns locais, a sua utilização fosse suspensa ou condicionada, por parte das autoridades locais. Noutros casos foram mesmo realizadas alterações à legislação local de forma a regularizar estes serviços. Em Portugal já existiu uma providência cautelar que levou à proibição de operação de um serviço *share economy* – a *Uber*.

Os serviços *share economy* apresentam por vezes problemas entre utilizadores, conhecidos como conflitos. Os serviços *share economy* não são responsáveis pelas atitudes que os prestadores de serviços ou clientes possam apresentar ao utilizar o serviço porém, se, por exemplo, um prestador de serviço for incompetente, no serviço que se disponibiliza para realizar, a imagem da empresa do serviço *share economy* será afetada. Por outro lado os prestadores de serviço também se devem sentir protegidos contra clientes mal-intencionados. Desta forma os serviços *share economy* apresentam, muitas vezes, mecanismo para resolver estes conflitos de forma a perceber quem os causou e tomar a melhor decisão com vista a evitar que os mesmos aconteçam.

A possibilidade de existirem conflitos leva muitas vezes as pessoas a desconfiar deste tipo de serviços que funcionam recorrendo à Internet e às aplicações móveis. Isto acontece porque a confiança é criada com a interação social entre as pessoas, e com estes serviços a interação é realizada com um computador ou um dispositivo móvel, levando que a desconfiança das pessoas nestes serviços seja um problema deste modelo de negócio.

Os serviços *share economy* possibilitam, em geral, formas de realizar os pagamentos dos seus serviços utilizando pagamentos eletrónicos. Esses pagamentos devem sempre respeitar as normas de segurança PCI DSS, de forma a evitar fraudes. Existem vários métodos de pagamento que podem ser utilizados, porém cada um apresenta as suas vantagens e desvantagens. Esses métodos de pagamento são implementados por diferentes serviços de pagamentos que fornecem uma abstração de todo o processo de pagamento.

Tendo em conta o que foi referido nesta secção procurar-se-á perceber como criar um serviço *share economy* no setor das entregas, utilizando as capacidades deste modelo de negócio e as restrições legais do mesmo. Por outro lado será importante, também, encontrar mecanismos de

resolução de conflitos e de criação de confiança juntando um serviço de pagamento seguro que seja adequado ao modelo de negócio, de modo a criar e fomentar a confiança dos utilizadores no serviço.

1.3 Objetivos

O objetivo principal é o desenvolvimento de uma aplicação móvel, como prova de conceito, para entregas de refeições, seguindo os princípios do modelo de negócio *share economy*, solucionando os problemas relacionados e explorando algumas das possíveis inovações associadas ao modelo.

Como exemplo escolheu-se uma aplicação que deverá permitir que os seus clientes encomendem refeições de um determinado restaurante. Após a encomenda, a aplicação deverá escolher o melhor estafeta para entregar a refeição na localização que o cliente indicou. Essa escolha deverá ter em conta vários fatores como a localização do estafeta, do restaurante e do local da entrega e a qualidade dos estafetas disponíveis.

Os pagamentos do serviço deverão ser realizados através da própria aplicação. Deste modo, é esperado que a aplicação utilize um serviço de pagamentos eletrónico que implemente os padrões de segurança e confidencialidade de dados.

Nestas aplicações a confiança é fundamental, e assim sendo, um dos objetivos será implementar alguns sistemas de criação de confiança, nomeadamente a verificação de identidade e criação de tabelas de classificação e credibilidade dos utilizadores através de avaliações efetuadas pelos clientes e até pelos seus pares.

Sabendo que este tipo de serviços está sujeito a conflitos entre os vários utilizadores, a aplicação deverá estar preparada para ajudar na resolução dos mesmos. Um exemplo é a utilização do GPS de forma a confirmar se o estafeta foi ao restaurante que o utilizador indicou.

1.4 Benefícios esperados com a solução do problema

Para os clientes é esperado que esta aplicação crie uma comodidade superior, dispondo de um serviço ainda não existente podendo receber encomendas com um serviço mais personalizado e com um custo competitivo em comparação com os serviços já existentes no mercado.

É também esperado que a aplicação possibilite aos prestadores a obtenção de mais rendimentos, assumindo o papel de estafetas, aumentando assim a qualidade de vida da população.

Por fim, espera-se que esta aplicação seja o impulsionador de novos serviços *share economy* em Portugal, de forma a trazer os benefícios deste modelo de negócio para outros setores da sociedade portuguesa.

1.5 Estrutura da dissertação

Esta dissertação esta organizada em seis capítulos. Este primeiro capítulo tem o objetivo de introduzir o problema e os objetivos desta dissertação.

O segundo e terceiro capítulo apresenta o estado da arte da *share economy* e de serviços e tecnologias utilizados na *share economy*. O capítulo número dois apresenta o funcionamento da *share economy*, analisando o seu modelo de negócio e apresentando exemplos de aplicações que implementam esse modelo. Já o terceiro capítulo apresenta tecnologias e serviços que estão presentes em implementações da *share economy*. Como é exemplo os serviços de pagamentos eletrónicos, sistemas de criação de confiança e de resolução de conflitos, os dispositivos móveis e as suas aplicações.

O quarto capítulo apresenta uma visão sobre os requisitos da solução a desenvolver e a sua arquitetura. Neste capítulo são abordadas perspetivas de solução para alguns dos problemas desta dissertação.

O capítulo referente à implementação, o quinto capítulo, faz uma apresentação da aplicação desenvolvida abordando os algoritmos e funcionalidades mais relevantes.

O sexto capítulo aborda os resultados, sendo apresentados os resultados dos testes que foram realizados à aplicação desenvolvida, acompanhados com as devidas conclusões desses testes.

O último capítulo apresenta uma conclusão do trabalho desenvolvido, incluído um resumo do trabalho elaborado, uma avaliação global da dissertação e ideias para desenvolvimento futuro.

Capítulo 2

Caracterização da *share economy*

O termo *share economy* é recente, e ainda pouco utilizado na literatura científica [Böck13]. No entanto, o mesmo é utilizado para identificar um modelo de negócio que permite, a um conjunto de indivíduos, realizar troca de bens ou serviços entre eles, através de serviços *peer-to-peer* [Böck13]. Essa partilha de serviços ou bens pode, em alguns casos, envolver uma compensação monetária ao prestador de serviço.

A partilha de bens e serviços sempre foi uma prática bastante comum, nomeadamente entre familiares e pessoas próximas, como vizinhos [Bird14]. Todavia, no final da primeira década do século XXI, o modelo de negócio *share economy* ganhou mais destaque. Esse acontecimento deveu-se a dois fatores: no início do século XXI a Europa e os Estado Unidos da América (EUA) mergulharam numa crise económica, que criou altas taxas de desemprego. Por outro lado, por essa mesma altura, a Internet e as chamadas novas tecnologias começaram a massificar-se [Inve14]. Os dois acontecimentos tornaram o modelo de negócio *share economy* mais apetecível. Se por um lado poderia ser uma forma de as pessoas conseguirem mais rendimentos, fazendo frente à crise, por outro, algumas empresas viram, neste modelo de negócio, uma oportunidade para aproveitarem as novas tecnologias que começavam a massificar-se na sociedade.

Em 2007, Brian Chomsky e Joe Gebbia, aperceberam-se desta oportunidade. Os dois amigos estavam a passar alguns problemas económicos, estando com dificuldades em conseguir pagar a renda da casa. Tiveram, então, a ideia de dar uso a três colchões de ar que possuíam, e alugá-los, de forma a conseguir algum rendimento extra. Deram o nome de *AirBed and Breakfast* a esse serviço de aluguer. Pelo serviço, que incluía o colchão e o pequeno-almoço, cobravam $80 por noite. Com esta ideia, ao fim de algum tempo, conseguiram ultrapassar os seus problemas económicos e, mais do que isso, tiveram a ideia de alargar este negócio criando uma rede global em que qualquer pessoa pode-se inscrever-se e alugar um quarto a um estranho. Esta ideia cresceu e, em 2009, tornou-se numa empresa, com o nome *Airbnb*, sendo muitas vezes comparada à cadeia de Hotéis *Hilton* [Frie13].

Os amigos Brian Chomsky e Joe Gebbia foram uns dos pioneiros a juntar esta ideia de negócio às novas tecnologias, nomeadamente à *Word Wide Web.* Começando, inicialmente, por ganhar dinheiro ao alugar bens subutilizados (os colchões) e, mais tarde ao abrir essa possibilidade para a sociedade, com a criação de uma plataforma *online*. É esta a ideia base da *share economy*, nos dias de hoje: conseguir rentabilizar algum bem subutilizado. Essa rentabilização pode ser feita através de venda, aluguer ou prestação de serviços.

2.1 Modelo de Negócio

As empresas que apostaram nesta área criaram plataformas *online* de forma a tentar melhorar um mercado já existente que, no seu ponto de vista, era ineficiente. Com as plataformas, tentaram criar um mercado *peer-to-peer* que aproxima o cliente do vendedor/prestador de serviço [Nguy14].

Estas empresas são recentes, mas, nos últimos anos, começaram a apresentar um grande volume de negócio. A revista económica norte americana *Forbes*, no início do 2013, estimou que o valor total das transações realizadas por *share economy,* nesse ano, seria na ordem dos três milhões e meio de dólares, representando um aumento na ordem dos 25% em relação ao ano anterior [Gero13].

Estas empresas seguem um modelo de negócio bastante semelhante entre elas. É importante perceber a forma como estas empresas escolhem os seus mercados e como conseguem tirar partido deles. Deste modo, será apresentado o modelo de negócio da *Airbnb,* com base na análise realizada por Giang Thu Nguyen em 2014, na sua tese de mestrado [Nguy14]. Fazendo também um paralelismo com o modelo de negócio genérico das *share economy*.

2.1.1 Modelo de negócio Airbnb

A *Airbnb* como já foi referido, foi fundada em 2009 por Brian Chomsky e Joe Gebbia. Fornecia um serviço que permitia qualquer pessoa alugar um quarto, colchão ou até um sofá a um estranho. O objetivo era com isso conseguir que turistas conseguissem uma estadia com menor custo do que num hotel convencional. Por outro lado os donos dos quartos, colchões ou sofás conseguiam rentabilizar algo que estava subutilizado.

Cliente alvo

Como em todas as empresas *share economy,* o *Airbnb* tem o objetivo de atingir dois segmentos diferentes de clientes:

- Pessoas que disponham de um espaço que podem alugar, de modo a conseguir rentabilizá-lo.

- Pessoas que procuram uma estadia a baixo custo.

Nas *share economies*, há clientes que prestam serviços e clientes que recebem serviços.

Na *Airbnb*, os prestadores de serviço são pessoas não ligadas, necessariamente, ao ramo da hotelaria, que desejam alugar algum espaço a outras pessoas. Por outro lado, os clientes que recebem os serviços são pessoas que procuram estadias de baixo custo, sendo geralmente turistas.

Embora a *Airbnb* não limite este segmento apenas a turistas, o seu *marketing* é mais direcionado a estes. O *marketing* da empresa apresenta muito o espírito de aventura e a ideia de conhecer novos locais e novas culturas.

Proposta de Valor

Tendo as *share economies* dois tipos de clientes, oferecem valor de forma distinta a cada um destes.

No caso dos prestadores de serviço, a proposta de valor apresenta-se na forma de divulgação dos produtos ou serviços do vendedor. Na *Airbnb*, o dono do espaço a alugar consegue ter uma projeção do seu anúncio, que de outra forma seria impensável, e consegue perceber quem são as pessoas que alugam o espaço pois, após uma estadia, há avaliação mútua. Posteriormente é criado um *rating* para cada utilizador (este ponto é, também, transversal a vários serviços *share economy*).

Por outro lado, o valor proposto aos clientes finais é a competitividade dos preços praticados nos serviços prestados, face ao mercado convencional. Na *Airbnb*, a empresa oferece mais, quando um cliente aluga um quarto obtém mais serviços, como por exemplo, guias de turismo. O serviço fornecido pela *Airbnb* promove, também, um maior intercâmbio de cultura, tanto para o turista como para o dono do espaço alugado.

Capacidades

As várias empresas com serviços *share economy* apresentam, em geral, plataformas digitais que permitem a comunicação dos prestadores de serviços com os clientes finais de uma forma eficiente, sendo esta a capacidade principal destas empresas. Porém, dependendo do contexto em que a empresa esteja inserida, poderá ter outras capacidades importantes.

A *Airbnb*, além da sua plataforma digital eficiente, possui também, uma capacidade de crescimento elevada. Entrou no mercado do turismo, onde já existe muita concorrência e, devido às estratégias utilizadas pela empresa, está a crescer e a atingir o topo do mercado.

Atividade

A análise das atividades é um ponto onde existe menos transversalidade entre as empresas de *share economy* pois este ponto da análise depende muito do contexto onde a aplicação está inserida. Todavia, existem pequenas atividades comuns, nomeadamente a avaliação dos utilizadores. É possível um utilizador avaliar outro utilizador com quem tenha efetuado uma transação, independentemente de ser cliente final ou prestador de serviço. Isto permite criar uma métrica da qualidade do utilizador.

A *Airbnb* focou-se muito, numa fase inicial, em conseguir ter *bloggers* e jornalistas a escrever sobre o serviço, de forma a conseguir apresentar a ideia ao mundo.

Para a *Airbnb* a qualidade do serviço é fundamental e está relacionada com a qualidade do espaço a ser alugado. Para conseguir ter um critério de qualidade, a empresa possui um sistema de feedback, em que algum cliente que tenha frequentado o serviço de aluguer possa classificar esse local.

Como a plataforma *online* da *Airbnb* é muito importante para os utilizadores, independentemente do papel destes, a empresa aposta bastante nesta capacidade. Possui um sistema de procura otimizado que tem em conta vários fatores, como por exemplo, as preferências do utilizador (que vai alugar um espaço). O sistema de pagamentos, embutido na plataforma, facilita a transação entre os intervenientes, permitindo, por vezes, recuperação de dinheiro em casos de fraude.

Parceiros

Este ponto também depende bastante do contexto do serviço porém, a maioria das empresas tem parcerias com empresas de pagamentos eletrónicos que lhes fornecem uma *gateway* de pagamento eletrónico. No caso em análise, além da parceria com uma empresa de pagamento eletrónico, a *Airbnb* possui uma parceria com uma empresa de apoio ao cliente.

Recursos e bens

Tal como nos pontos anteriores, a transversalidade entre as várias empresas *share economy* deste ponto de vista é baixa, por ser muito específico. No entanto, um dos principais recursos destas empresas é a equipa de programadores que desenvolve e dá suporte às plataformas *online*.

A *Airbnb* conta, também, com equipas para contacto com grandes clientes, para resolução de conflitos entre clientes e resolução de problemas legais, como pagamento de impostos.

Relacionamento com o cliente

O relacionamento com o cliente é fundamental neste negócio. A empresa em análise, tal como todas as empresas deste modelo de negócio, tentam manter um contacto próximo com o cliente.

Canais de distribuição

Para maior proximidade com o cliente, as empresas tentam aproveitar ao máximo a tecnologia disponível. Estas empresas apostam no seu portal *Web*, nas suas aplicações móveis e redes sociais. Algumas, como é o caso da *Airbnb,* possuem um *blog* corporativo onde apresentam as novidades do serviço.

Aspetos financeiros

As receitas destas empresas surgem essencialmente da cobrança de pequenas taxas na utilização das suas plataformas. A *Airbnb* cobra entre seis a doze por cento à pessoa que efetua a reserva e três por cento ao dono do local a alugar.

Por outro lado, as receitas estão relacionadas com o custo de manutenção e desenvolvimento das plataformas que estas empresas possuem, *marketing* e apoio ao cliente.

2.1.2 Problemas legais

As empresas *share economy* apresentam algumas questões legais que não estão bem clarificadas. As questões mais relevantes são, o pagamento de impostos, a concorrência desleal e a segurança dos clientes.

Existem vários casos de empresas proibidas de fornecer o seu serviço em algumas cidades ou países devido a questões legais que não estão bem definidas. A *Uber*, uma empresa *share economy*, que funciona como um sistema de táxis *on-demand*, já enfrentou vários casos de proibição. Em dezembro de 2014, foi proibido em Espanha, após uma queixa, por parte de uma empresa de taxistas, de concorrência desleal [Bbc14a]; também em dezembro de 2014, a mesma empresa foi proibida em Nova Deli, Índia, após um condutor *Uber* ter violado uma jovem que usufruía do serviço [Bbc14b], mais recentemente a *Uber* foi, também, proibida de operar em território português após uma providência cautelar apresentada pela ANTRAL [Banc15]. Esta decisão levou também ao bloqueio do *website* da empresa em Portugal [Séne15]. A *Airbnb* também tem alguns problemas legais, um deles é a existência de pessoas que disponibilizam casas completas para aluguer em vez de um simples quarto, criando concorrência desleal com o sector hoteleiro, já que estes têm uma carga fiscal superior. Para evitar isto, a cidade de São Francisco nos EUA, criou um conjunto de normas, onde se inclui uma relativa ao aluguer de casas

completas. Só é legal disponibilizar uma casa completa para ser alugada, pela *Airbnb*, durante noventa dias por ano [Marc14].

Nas duas empresas utilizadas como exemplo, em alguns dos casos, o motivo que levou à proibição foi a concorrência desleal. No caso da *Uber*, os condutores não pagam taxas que os taxistas normais pagam [Sund12], tendo vantagem competitiva. Da mesma forma também se levanta a questão de fuga aos impostos, por parte do condutor pelo não pagamento dessas taxas. O caso da *Airbnb* é semelhante, o sector hoteleiro tradicional paga taxas para o hotel se manter aberto, enquanto um utilizador de *Airbnb* não [Sund12].

A questão de segurança é também muito importante. Por exemplo, a *Uber* não cumpre os mesmo requisitos que um serviço de táxis tradicional. E o mesmo se aplica à *Airbnb* [Sund12].

Estas questões têm tido várias interpretações diferentes, mas existe um consenso, é uma área de negócio benéfica para a sociedade necessitando de nova regulamentação de forma a tornar os serviços mais seguros e mais justos para com os serviços tradicionais.

2.1.3 Aplicações share economy

Já foram apresentados alguns exemplos de aplicações *share economy*, como a *Airbnb* e a *Uber* mas este modelo de negócio abrange muitas mais áreas distintas e mais possibilidades.

Transportes

Nesta área, o tipo de serviço mais conhecido é o táxi *on-demand* que permite aos clientes contratar um condutor para o transportar de forma semelhante a um táxi convencional. A vantagem está presente na possibilidade de poder chamar um condutor ao local onde se encontra, através de uma aplicação móvel. Um exemplo deste serviço é o já referido *Uber*, mas existem outros semelhantes como por exemplo, o *Lyft*.

Embora o serviço de táxi *on-demand* seja o mais conhecido, existem outras formas de aplicar a *share economy* na área dos transportes. A empresa *Zip Car* é um serviço de aluguer de automóveis. Todavia apresenta uma diferença, em relação ao serviço convencional, a entidade que aluga o automóvel. Os automóveis pertencem a cidadãos e não a empresas de aluguer. Estando essas pessoas a alugar os seus automóveis pessoais.

Ainda na área de transportes, existem também empresas a criar aplicações *share economy* na área dos estacionamentos. Aplicações como o *Parking Panda* e o *SpotHero* permitem alugar garagens a desconhecidos. Já a aplicação italiana *MonkeyParking* permite qualquer condutor que tenha um lugar de estacionamento o leiloar a outros condutores.

Viagens e hotelaria

Na área da hotelaria, os serviços que existem são semelhantes à *Airbnb*. A *Airbnb* já foi analisada em detalhe na secção 2.1 assim sendo, de uma forma resumida a empresa fornece um

serviço que permite o aluguer de espaços (quartos ou casas) a estranhos. Esta empresa não está sozinha neste mercado, existem empresas, como a *Tansler*, que criam concorrência no mercado.

Serviços ao domicílio

Os serviços ao domicílio são uma área bastante vasta. Nesta área, existe uma empresa que se destaca, *TaskRabbit*, devido à sua abrangência. O *TaskRabbit* permite encontrar pessoas para realizar determinadas tarefas, como por exemplo, limpeza de casa ou pequenos trabalhos de *bricolage*.

O *TaskRabbit* é um serviço muito genérico que abrange qualquer tipo de tarefa mas, existem sistemas mais específicos. A aplicação *Swifto* é utilizada para encontrar pessoas dispostas a passear animais de estimação. Um outro exemplo é a aplicação *urbansitter*, que encontra amas para cuidar de crianças.

Saúde e beleza

Na *share economy*, ao nível de aplicações de saúde e beleza existem serviços relacionados com massagens, maquilhagem e *manicure*. Existem algumas aplicações, como a *Zeel* e a *Vensette*, que disponibilizam este tipo de serviços.

Também na área da saúde e do exercício físico, a aplicação *Mindbody* permite a profissionais anunciar uma aula, de forma a facilitar a procura dessas mesmas aulas.

Restauração

A área da restauração conta com serviços *share economy* para encomendas particulares. Cozinheiros, profissionais ou não, recebem os pedidos e confecionam a refeição encomendada, entregando a mesma nas residências dos seus clientes. A aplicação *Munchery* é um exemplo de uma aplicação que fornece este serviço.

Entregas

A área das entregas é muito abrangente e é um dos focos desta dissertação. Desse modo, serão analisadas algumas aplicações que fornecem serviços *share economy* na área das entregas.

O serviço *Zipments* é semelhante ao serviço de correios convencional ou seja, um utilizador escolhe o que quer enviar e para onde. O que o diferencia do serviço convencional é a forma como é calculado o preço a pagar. O utilizador define um preço a pagar pelo serviço, e os estafetas podem aceitar, recusar ou fazer contrapropostas, criando um sistema de leilão. O utilizador pode depois escolher o estafeta que irá realizar a entrega.

Já a *Postmates* é um serviço que conta com ligações a lojas e a restaurantes. Com a aplicação, o utilizador pode escolher o que quer encomendar da loja ou restaurante e definir a hora de entrega. A entrega é realizada por um estafeta, e durante o tempo de entrega o utilizador pode acompanhar a localização do estafeta, de forma a perceber o tempo que resta até chegar à sua residência.

A aplicação *WunWun* é bastante semelhante à *Postmates*. O utilizador escolhe o produto a comprar, a hora e o local de entrega. A diferença entre os dois serviços é o modo de pagamento. A *WunWun* é gratuita numa determinada zona geográfica, embora os utilizadores sejam incentivados a premiar os estafetas com donativos.

A *Dorman* pertence à área das entregas mas numa vertente diferente. A *Dorman* tenta resolver o problema que existe quando se recebe uma encomenda e não se está em casa para a receber. Funciona como um depósito de encomendas ou seja, as encomendas são entregues na *Dorman* e, como este serviço possui estafetas com horários mais flexíveis, é possível entregar a encomenda ao utilizador à hora que lhe é mais conveniente. Para utilizar este serviço, o utilizador tem que indicar a morada de entrega da *Dorman*, juntando um identificador de utilizador, aquando da compra *online* [Pere14].

As aplicações *Shyp* e *Shipster* são muito semelhantes. O objetivo destas aplicações é permitir ao utilizador enviar encomendas para qualquer lado de uma forma muito simples. Basta fotografar o item a enviar e o estafeta dirige-se à localização do utilizador para tratar da encomenda. Estas empresas tratam de todo o processo de entrega, desde o empacotamento até à entrega propriamente dita.

A aplicação *Instacart* permite que o utilizador escolha produtos de determinadas lojas, de forma semelhante ao *WunWun*, que depois serão entregues na sua residência por um estafeta.

A Tabela 2.1 apresenta uma análise a estas várias aplicações na área das entregas.

Tabela 2.1: Comparação entre várias aplicações de entregas

	Zipments	***Wun Wun***	***Postmates***	***Dorman***	***Shyp***	***Shipster***	***Instacart***
Método de Pagamento	*Stripe*	*Stripe*	-	-	-	-	*Stripe*
Rating e Ranking	Sim	Não	Sim	Não	Não	Não	Não
Acompanha-mento em tempo real	Sim	Sim	Sim	Sim	Sim	Sim	Sim
Sistema de leilão	Sim	Não	Não	Não	Não	Não	Não
Localização	Austrália, Canada e EUA	EUA	EUA	EUA	EUA	EUA	EUA

Plataformas	*Web* *iOS* *Android*	*Web* *iOS*	*Android* *iOS*	*iOS*	*iOS*	*iOS*	*Web* *iOS* *Android*

2.2 Resolução de Conflitos

Por vezes na utilização de aplicações *share economy* existem conflitos entre os vários intervenientes. Esses conflitos devem ser resolvidos pela empresa detentora do serviço de forma a evitar que esses conflitos possam estragar a imagem da empresa. Para isso, as aplicações devem estar preparadas para perceber o que realmente aconteceu, de forma a resolver o conflito o mais justamente possível.

As empresas não divulgam muito a forma como os conflitos são solucionados, porém, algumas dessas técnicas de resolução de conflitos são visíveis ao utilizar as aplicações dessas empresas ou são descritas por utilizadores em fóruns na *web* para esse efeito [14a, 14b, 14c]. A *Uber* guarda as localizações dos condutores de forma a perceber se realmente esse condutor transportou o passageiro e utiliza essa informação para proteger o condutor. Se o condutor chegar ao local onde o passageiro devia estar, esperar durante 6 minutos e este não aparecer, o condutor é recompensado (5 dólares) e pode abandonar o local.

Segundo relatos de utilizadores da *Uber* [14a, 14b, 14c], a tendência de resolução de conflitos por parte da empresa não é completamente isenta, dando, em caso de situações menos claras, mais vezes razão aos passageiros do que aos condutores.

Em geral, as empresas tentam resolver estes assuntos internamente sem causar muito alarido, pedindo aos seus utilizadores para relatarem os problemas ocorridos para serem resolvidos posteriormente. O *TaskRabbit* na secção de FAQ do seu fórum dá mesmo esse conselho aos seus utilizadores [Task14].

2.3 Confiança

Gambetta [Gamb88] define a palavra confiança como sendo a probabilidade de o *sujeito A* acreditar que o *sujeito B* realize uma ação da qual o bem-estar do *sujeito A* depende. Esta definição encaixa-se perfeitamente nos serviços *peer-to-peer*. Nestes serviços, a confiança depende da crença que um interveniente tem em que o outro interveniente irá cumprir a tarefa que lhe foi proposta.

A confiança entre pessoas é criada com a interação entre elas, de onde o cérebro humano cria a sensação de confiança através de vários fatores como a aparência, o primeiro contacto e o resultado se outros contactos com essa pessoa. No mundo *online* não existe acesso às mesmas sensações, que o cérebro humano utiliza para criar a confiança no mundo *offline,* por isso é importante aproveitar as funcionalidades que a Internet disponibiliza de forma a criar outras sensações que substituam essas [JøIB07].

2.3.1 Métodos de criação de confiança em aplicações *Share Economy*

Como qualquer serviço que utilize transações de dinheiro ou aceda a dados confidenciais dos utilizadores, os serviços *Share Economy* também têm de transmitir uma sensação de confiança para os seus utilizadores. Sabendo disso, as aplicações já existentes no mercado implementam alguns mecanismos para criar a confiança necessária nos utilizadores dos seus serviços.

Verificação de identidade

Uma forma de criação de confiança encontrada pelas aplicações *share economy* com serviços *peer-to-peer* foi a identificação de cada utilizador do sistema. É importante que pelo menos o prestador de serviço seja bem identificado e para isso, a maioria das aplicações desta área obriga à uma validação da identidade dos seus prestadores de serviço.

A *Airbnb*, para validar a identidade, utiliza um cruzamento de dados pessoais com os dados de redes sociais. Os dados pessoais são obtidos através de uma fotografia do seu cartão de identificação civil. Com os dados das redes sociais e do cartão verifica se os dados que o prestador de serviço indicou no registo coincidem.

A *Airbnb*, para analisar a fotografia do documento de identificação civil, usa a *framework Jumio* que permite ler documentos utilizando uma câmara, seja esta de um dispositivo móvel ou de um computador. Esta *framework* tem capacidades para interpretar cartões de crédito, cartões de identificação civil, licenças de condução e passaportes de vários países. Pode ser utilizada para preenchimento de dados de uma forma mais rápida, como por exemplo preencher o formulário de uma página *web* que necessite das informações de um cartão de crédito, a fotografia do cartão é suficiente para a *framework* interpretar os dados necessários para preencher o formulário. Em Portugal, o *Jumio* interpreta o Cartão de Cidadão, o Passaporte Português e a Carta de Condução [Jumi14].

Por sua vez, o serviço *Uber* obriga o condutor a apresentar o seu documento de identificação civil, carta de condução sem registo de infrações nos últimos anos, registo do seu veículo com a respetiva apólice de seguro e ainda o seu registo criminal. O objetivo da *Uber* é ter bons condutores, sem antecedentes criminais e com automóveis seguros.

Outras empresas *share economy* optam por fazer entrevistas aos seus futuros prestadores de serviço e, em alguns casos, darem alguma formação. Todavia, de forma semelhante à *Uber*, a maioria opta por se informar do passado criminal dos candidatos a prestadores de serviço.

Os clientes finais não têm um controlo tão apertado como os prestadores de serviço, por dois motivos. Os clientes devem poder aceder, de uma forma rápida e simples ao serviço e por outro lado os prestadores de serviço são a imagem da empresa sendo esse o motivo do controlo mais apertado. Deste modo, aos clientes, em geral, são pedidos alguns dados pessoais como o nome, telefone e uma conta de pagamento (cartão de crédito ou conta *PayPal*) sendo verificado se a conta de pagamento é válida e se o número de telefone pertence ao utilizador.

Sistemas de classificação

Uma outra forma de aumentar a confiança nos utilizadores é utilizar sistemas de classificação – *ratings* e *rankings* – de utilizadores com dados de avaliações efetuadas pelos pares.

Muitos dos sistemas *share economy* recorrem à avaliação dos pares de forma a criar uma classificação dos vários utilizadores. Um dos sistemas de avaliação mais utilizado é o que o serviço *eBay* apresenta, por pontuações. Esse sistema permite a um utilizador classificar outro depois de efetuar uma transação com ele. Essa classificação pode ter um de três valores: positivo (1), neutro (0) ou negativo (-1); juntando a essa classificação um campo de apreciação qualitativa [JøIB07]. É criado um *rating*, calculado com recurso a todas as classificações. Porém, este sistema apresenta algumas desvantagens como por exemplo, a existência de classificações negativas (10) de um utilizador que contrabalancem com classificações positivas (100) e o resultado final (90) seja igual a um utilizador com classificações apenas positivas (90). Embora o *rating* dos utilizadores seja igual, o utilizador que não obteve nenhum *feedback* negativo apresenta mais confiança [JøIB07], mas com este sistema isso não é devidamente refletido na classificação. Uma forma de evitar este problema é aumentar o domínio da avaliação, em vez de apenas três opções utilizar 5 ou até 10.

Muitas empresas *share economy* utilizam sistemas baseados no do *eBay* adaptando ao seu contexto. A *Uber* utiliza este sistema mas acrescenta alguns parâmetros como, a taxa de conclusão de transportes efetuados com sucesso ou, a taxa de cancelamentos que o condutor apresenta. Já no caso do *Lyft*, utiliza um sistema semelhante ao *eBay* mas com uma escala de 5 valores (estrelas), existindo relatos [Iyer14] de utilizadores que, após ter classificado um condutor com três estrelas terem sido contactados pela empresa de forma a perceber o que tinha corrido mal na viagem.

O sistema utilizado pelo *eBay* só pode ser visualizado pelos utilizadores do *eBay* [Gree00] porém essa informação seria útil se pudesse ser consultada noutros serviços onde esse utilizador também está registado. Essa questão foi solucionada pela *TrustCloud*, criando um serviço que compila dados de classificações de vários serviços.

O sistema *TrustCloud* encontra-se ainda numa fase *Beta* mas já é utilizada por pequenas empresas estando apenas disponível nos EUA. Este sistema faz um cruzamento de dados entre várias contas que o utilizador possui em diferentes serviços. Essas contas podem ser redes sociais, *websites* de perguntas e respostas, como o *Stack Overflow*, ou aplicações *share economy*, como *Airbnb* e *TaskRabbit*. Ainda faz verificações de contas de correio eletrónico, telefone e morada.

O sistema apresenta um pouco do conceito de *gamification*, quanto mais dados o utilizador fornecer melhor é a sua classificação. Os dados que o utilizador fornece geram um *TrustCard*

Figura 2.1: *TrustCard*

como o da Figura 2.1. Este cartão contém informação do utilizador como a pontuação de confiança que lhe foi atribuída. O utilizador pode apresentar esse *TrustCard* mesmo em serviços que não estejam ligados à *TrustCloud*, podendo guardar o cartão como imagem ou utilizar um *script JavaScript* com o mesmo.

Capítulo 3

Serviços e tecnologias para as *share economies*

As atuais aplicações de *share economy* necessitam da utilização de alguns serviços e recorrem a tecnologias muito relacionadas coma utilização da *web* e da mobilidade. Contam-se entre estes serviços e tecnologias os pagamentos eletrónicos, a utilização de dispositivos móveis e suas aplicações clientes nativas, e ainda as tecnologias adequadas à criação de servidores e serviços *web*. Dessa forma serão abordados, neste capítulo, alguns desses serviços e tecnologias.

3.1 Pagamento eletrónicos

Desde a afirmação da Internet que o paradigma de pagamentos eletrónicos tem sofrido alterações, tornando-se cada vez mais comum e frequente. O grande sucesso deste modo de pagamento deve-se a algumas vantagens que apresenta face aos demais. Os utilizadores têm uma sensação de segurança, sendo os serviços acessíveis, rentáveis e eficientes [Cott17, KoPA08, LiPW06, TsSt05].

Os pagamentos eletrónicos podem ser classificados em cinco categorias [DaGr07, Guan03, LNCL03, Mark01, Schn52]:

- ***Electronic-cash***: Também conhecido com *e-cash* [Chou04, Jews01, Wrig02], foi criado com o objetivo de substituir o cartão de crédito no pagamento de bens e serviços na Internet [Mark01]. É um método de baixo custo, ideal para pequenos pagamentos [Kim05, LNCL03].

- ***Cartões pré-pagos***: São cartões emitidos com um valor pré-determinado. São utilizados, em geral, como presentes [Knib02].

- ***Cartão de crédito***: São a forma de pagamento mais utilizada nos pagamentos eletrónicos [Chou04, Hsie02].

- ***Cartão de débito***: A grande diferença entre os cartões de crédito e de débito é a forma como o pagamento é processado. Com os cartões de débito a transferência é feita diretamente da conta do dono do cartão [Mark01].

- ***Cheques eletrónicos***: Uma instituição estabelece eletronicamente as transações entre o banco do comprador e o banco do vendedor na forma de um cheque eletrónico [KTSK10].

O comércio eletrónico envolve sempre pelo menos duas entidades. Essas entidades podem ser de três tipos diferentes [RBZJ02]:

- Empresas

- Consumidores

- Governo

Dependendo dos intervenientes, as transações tem designações diferentes. As transações entre empresas são conhecidas por *business-to-business* (B2B), já quando a transação ocorre entre uma empresa e um consumidor é chamada *business-to-consumer* (B2C) ou *consumer-to-business* (C2B), dependendo da direção da transação. Quando a transação é realizada entre dois consumidores tem o nome de *consumer-to-consumer* (C2C), *peer-to-peer* (P2P) ou *person-to-person* (P2P) dependendo da literatura [King12a, RBZJ02].

Na Figura 3.1 está representada uma classificação dos sistemas de pagamento eletrónico. Na figura é possível verificar que as transações entre empresas são realizadas, essencialmente, por cheque eletrónico. Por outro lado, nas transações de clientes com outros clientes ou com uma empresa são utilizados sistemas mais variados, podendo ser *Electronic Cash*, cartões de pré-pagamento, e cartões de crédito ou débito [KTSK10].

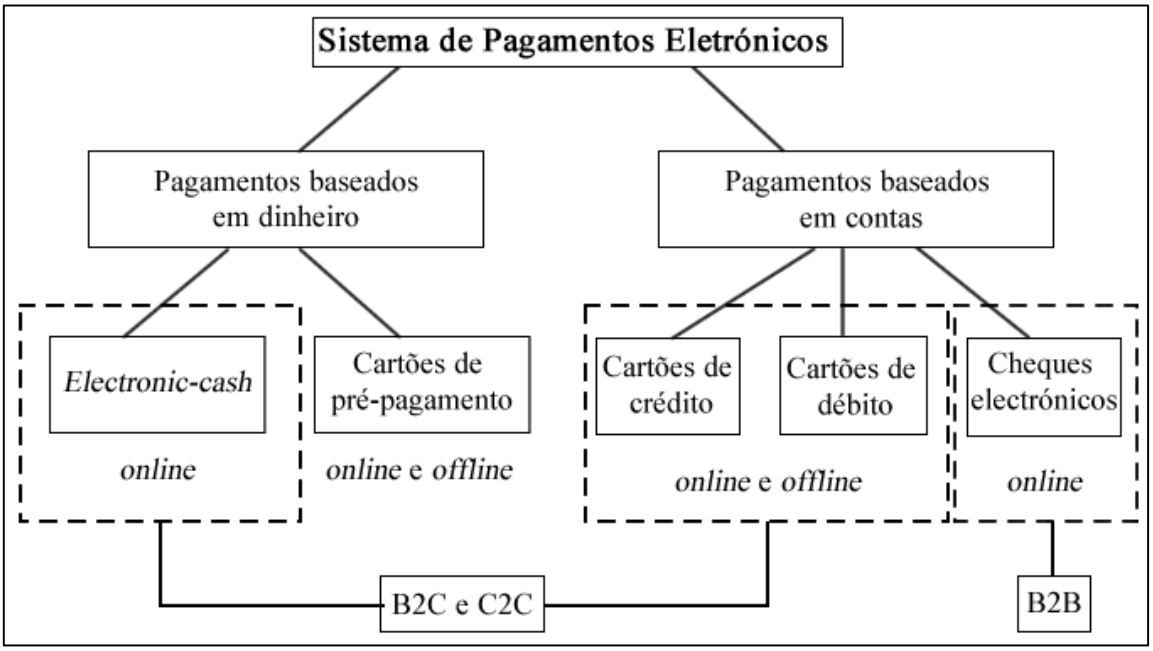

Figura 3.1: Sistema de Pagamentos Eletrónicos

Os pagamentos nas *share economies* são em geral realizados entre consumidores ou seja, C2C. Em alguns casos, existem também pagamentos à empresa, sendo pagamentos C2B.

3.1.1 Pagamentos C2C

Os pagamentos C2C criam um fluxo de dinheiro entre dois consumidores. Esse fluxo é criado por uma entidade responsável pela transferência desse dinheiro, de uma conta bancária para outra. A plataforma mais conhecida com esta capacidade é o *PayPal*.

O *PayPal* surgiu em 1998 e em 2014 possuía 156.9 milhões de contas ativas em mais de cem países [Stat14]. O *PayPal* permite que um utilizador, criando uma conta, associe um cartão de crédito e, após algumas validações de segurança, o utilizador pode transferir o dinheiro para qualquer pessoa que possua um endereço de correio eletrónico [Gonz04].

3.1.2 *MarketPlace*

Como já foi referido, as *share economies* utilizam sistemas de pagamento C2C e em alguns casos C2B. Para as empresas *share economies* o ideal seria um método de pagamento misto, onde uma percentagem do pagamento vai diretamente para a empresa e outra para o prestador de serviço. A essa forma de pagamento chama-se *MarketPlace.*

O sistema de pagamentos *MarketPlace* é bastante recente. Este modo de pagamento eletrónico tem três entidades: o consumidor, o prestador de serviço de pagamento e o mercado (empresa).

Quando os serviços *share economy* começaram a aparecer, a melhor forma de efetuar os pagamentos era recorrer ao *PayPal* mas, os consumidores eram obrigados a criar uma conta e o serviço apresentava custos extras, sendo do seu desagrado [Just12].

Nos últimos anos, apareceram novos prestadores de serviço, de pagamentos, que aplicam o conceito de *MarketPlace*. A Tabela 3.1 apresenta uma análise desses serviços.

Na Tabela 3.1 é possível observar que metade dos serviços apresentados só podem ser utilizados por empresas sediadas nos EUA. Por sediada entende-se que a empresa que utiliza o serviço de pagamento tem de ter um número de identificação fiscal e uma conta bancária desse país.

Dos serviços apresentados, só o serviço alemão, *PayMill*, permite ser utilizado por empresas sediadas na Europa. Todavia, em agosto de 2014, a Comissão Europeia [Comm14] anunciou a migração do sistema de pagamentos para o *Single Euro Payments Area* (SEPA), em português, Área Única de Pagamentos em Euros.

Tabela 3.1: Comparação entre vários serviços de pagamento *MarketPlace*

	Stripe	***Brain Tree***	***Pin Payments***	***PayMill***	***WePay***	***Balance***
Método de Pagamento	Cartão de crédito Cartão de débito *Apple Pay*	Cartão de crédito Cartão de débito *Apple Pay*	Cartão de crédito	Cartão de crédito Cartão de débito	Cartão de crédito	Cartão de crédito
Suporte dispositivos Móveis	Sim	Sim	Não	Sim	*iOS*	Sim
Preço por transação[1]	2,9% + 30¢	2,9% + 0,30 €	2,6% + 30¢	2,95% + 0,28 €	2,9% + 30¢	2,9% + 30¢
Países onde pode ser sediado	EUA, UK, Irlanda, Canada e Austrália	EUA	Austrália	Países Europeus	EUA	EUA
Países onde pode ser utilizado	Qualquer país	130 Países	Qualquer país	Qualquer país	EUA	Qualquer país

[1] O valor em percentagem é aplicado ao valor da transação

Segundo o Banco de Portugal [Banc13], a SEPA permite a particulares, empresas e organismos públicos efetuarem pagamentos utilizando uma única conta bancária localizada em qualquer país da União Europeia, Islândia, Liechtenstein, Mónaco, Noruega ou Suíça.

Tendo em conta este novo sistema, é previsível que em breve possam aparecer novos serviços como o *PayMill* ou que outros serviços já existentes comecem a suportar mais países europeus.

3.1.3 Processamento de um cartão de crédito

O processamento de um cartão de crédito é um processo complexo que é caracterizado por várias fases [Cred13] e várias entidades diferentes que processam os dados.

A primeira entidade é o *dono do cartão*, esta entidade tem na sua posse todos os dados do cartão.

O *dono do cartão* comunica só com a entidade *comerciante*, que é uma entidade que suporta pagamento com recurso a cartões de crédito. Esta entidade, o *comerciante*, deve cumprir as normas PCI DSS.

Por sua vez o *comerciante* apenas comunica com o *adquirente*. Esta entidade é responsável por adquirir os créditos dos *comerciantes*. É a esta entidade que os comerciantes devem transmitir os dados relativos às transações. Os *adquirentes* impõem comissões designadas por *interchange fee* [Banc09].

Por fim os adquirentes contactam com as *redes de cartões* e com os *emissores*. As redes de cartões são as *companhias dos cartões* (*Visa*, *MasterCard*) que funcionam por vezes como

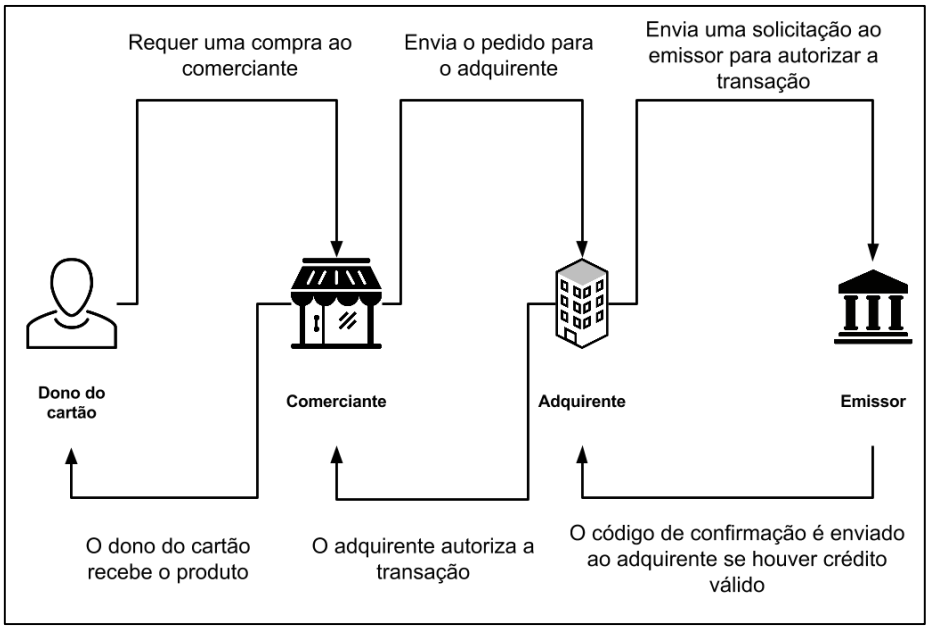

Figura 3.2: Autorização de pagamento

intermediários entre os *adquirentes* e os *emissores*. Já os *emissores* são entidades financeiras que emitem os cartões de crédito [Cred13].

O processo de pagamentos pode ser dividido em quatro fases. A primeira fase é a *Autorização*. Este processo acontece quando um *dono do cartão* pretende comprar um produto ou pagar um serviço. A Figura 3.2 apresenta o fluxo desta fase do processo.

Nesta fase o *dono do cartão* requer ao *comerciante* a compra do produto. Este por sua vez envia o pedido de comprar com os dados do cartão para o *aquirente*. O *aquirente* valida juntamente com o *emissor* os dados do cartão. Se for válido o *adquirente* autoriza a transação e o *comerciante* entrega o produto ao *dono do cartão*. O próximo passo é o *Batching* que é executado pelo *Comerciante* todos os dias. A Figura 3.3 apresenta o funcionamento deste passo.

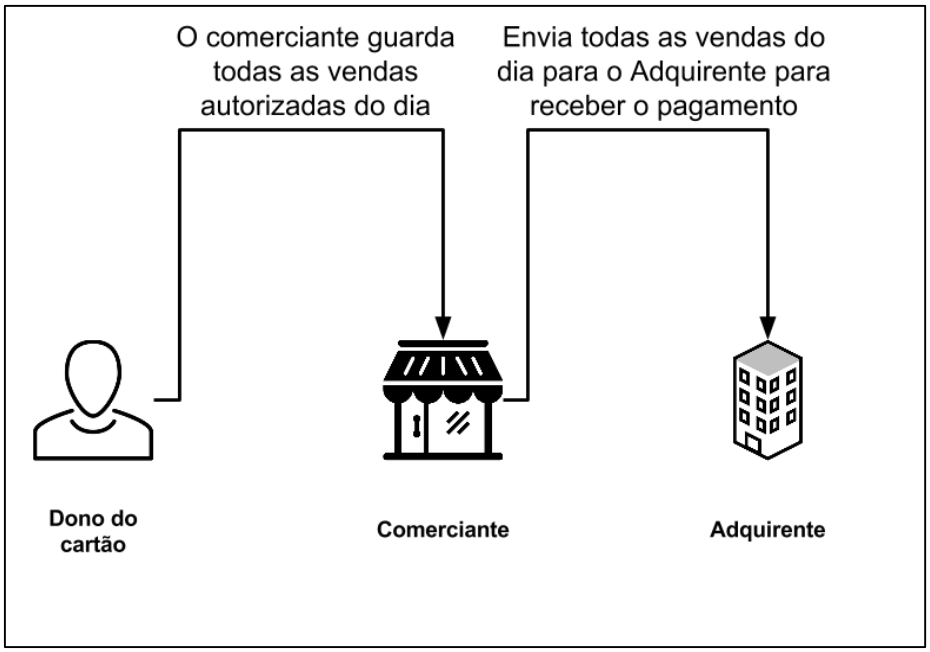

Figura 3.3: *Batching*

Neste passo o *comerciante* deve guardar todas as vendas que autoriza durante o dia, e no final do mesmo deve requerer os pagamentos ao *adquirente*.

O próximo passo tem o nome de *Clearing*, neste passo o adquirente reencaminha os pedidos de pagamentos, que o *comerciante* lhe enviou, para *a companhia dos cartões*. Esta por sua vez reencaminha o pedido para o *emissor*. De seguida são subtraídas as taxas e comissões e o montante é transferido para o *adquirente*. A Figura 3.4 demostra este processo.

O último passo é o *Funding*, neste passo o *adquirente* subtrai uma comissão e envia o

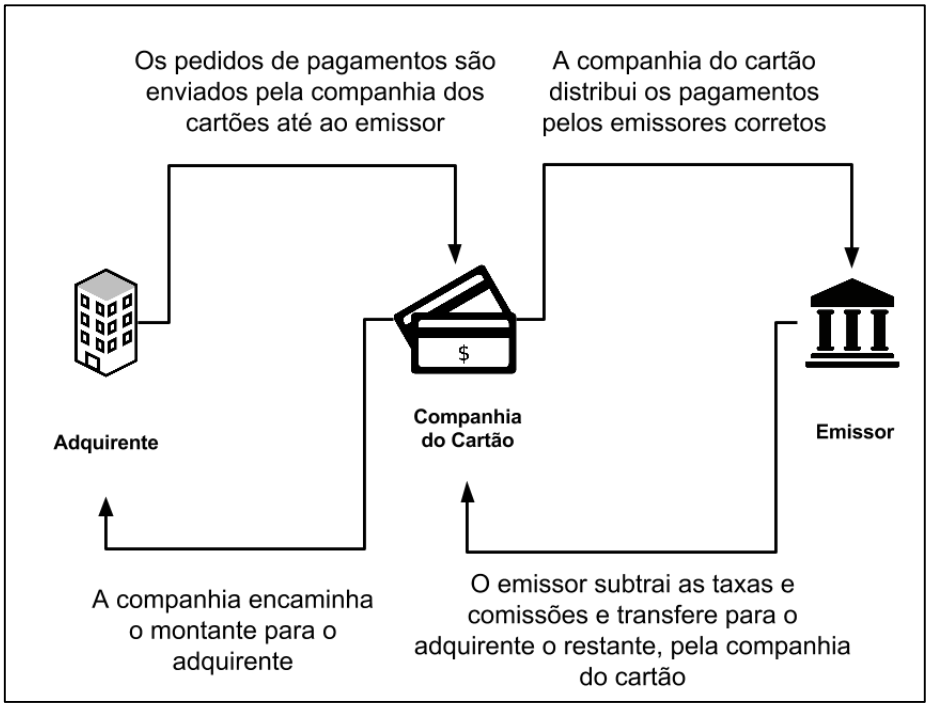

Figura 3.4: Clearing

restante para o *comerciante*. Neste passo é também realizada a cobrança ao *dono do cartão*. A Figura 3.5 apresenta este processo.

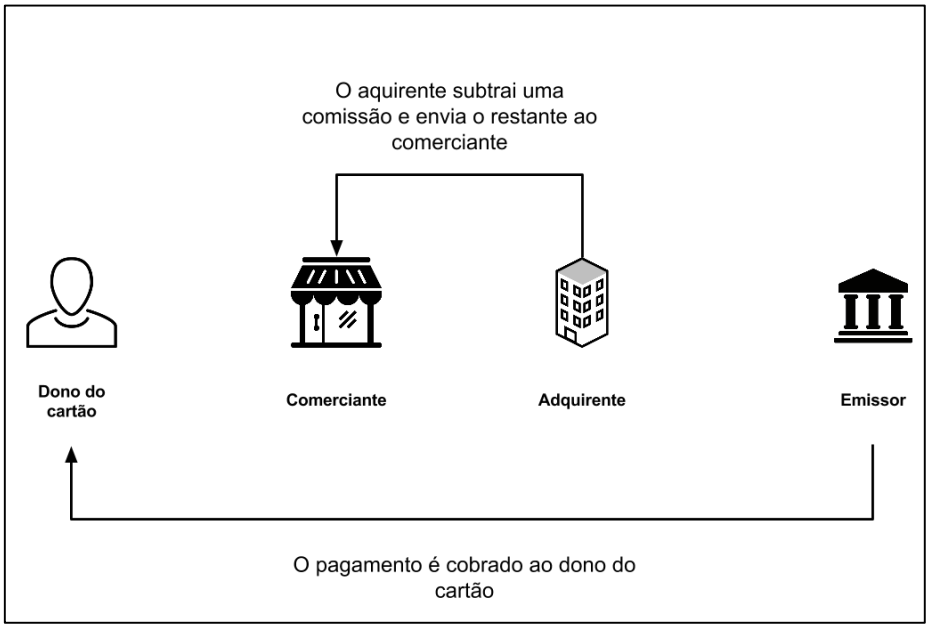

Figura 3.5: *Funding*

3.1.4 Pagamento eletrónico em dispositivos móveis

Os pagamentos com dispositivos móveis estão a ganhar um grande destaque na sociedade. Em 2011, foram realizados mais pagamentos utilizando dispositivos móveis do que com cheques

[King12b]. Este tipo de pagamentos inclui compras de aplicações (para os dispositivos móveis), pagamento com recuso a SMS, pagamentos de transferência direta de saldo do telemóvel, pagamentos em moeda virtual e pagamento P2P [King12a].

O serviço *Boku* permite efetuar pagamentos utilizando o saldo do telemóvel. O fluxo de funcionamento deste sistema é relativamente simples:

1. Escolhe o produto ou serviço a comprar, no *website* ou aplicação móvel;
2. Introduz o número de telemóvel;
3. Recebe uma SMS no número de indicou com um código e introduz no *website/*aplicação para confirmar a compra.
4. O valor da comprar é descontado no saldo do telemóvel do utilizador.

Para funcionar de forma correta, o *Boku* necessita de parcerias com as operadoras de telecomunicações de modo a poder processar as transferências monetárias. Atualmente, o serviço tem parcerias com duzentas e cinquenta empresas de telecomunicações em todo o mundo e entre elas, as três operadoras nacionais, *MEO*, *NOS* e *Vodafone* [Boku14].

Um outro serviço na área de pagamentos com dispositivos móveis é o *Google Wallet*, lançado em 2011. Este serviço permite que um utilizador possa armazenar, no seu *smartphone* (*Android* ou *iOS*), os seus cartões de crédito, para utilizá-los em pagamentos numa futura compra.

O *Google Wallet* possibilita a realização de compras *online* ou em lojas físicas. Quando utilizado em lojas físicas, o pagamento é efetuado recorrendo a tecnologias de comunicação NFC, estando limitada a *smartphones* com essa tecnologia. De forma semelhante ao *PayPal*, o *Google Wallet* permite transferir dinheiro entre utilizadores todavia, esta funcionalidade apenas está disponível para utilizadores norte-americanos. As restantes funcionalidades estão disponíveis para um grande número de países [Goog14].

A *Apple*, por sua vez, em 2014 lançou o *Apple Pay*. É um serviço semelhante ao *Google Wallet* e funciona apenas em *iPhones* e *iPads* mais recentes, usando algumas características do *hardware* desses dispositivos (nomeadamente do *iPhone 6*) para adicionar algumas medidas de segurança extra como por exemplo, a utilização do *Touch ID* que utiliza a impressão digital para o utilizador validar um pagamento em loja. Este serviço encontra-se somente disponível nos EUA.

O *Apple Pay* diferencia-se do *Google Wallet* no que diz respeito ao armazenamento de dados. Ao contrário da Google a Apple não armazena os dados dos cartões nos seus servidores utilizando o sistema de *tokenização*. Este sistema substitui os elementos chave, como por exemplo o número do cartão e o CVV, por *tokens* gerados pela entidade emissora do cartão [Ultr15].

3.1.5 Outros sistemas de pagamento eletrónico

Existem outros sistemas de pagamento que, embora não sejam totalmente direcionados aos dispositivos móveis, também são compatíveis com os mesmos, um exemplo é o serviço *easypay*.

Este serviço de pagamento, criado por uma empresa portuguesa, permite que o consumidor possa efetuar pagamentos de três formas diferentes: por multibanco, utilizando uma referência de pagamento, por catão de crédito ou transferência bancária.

Para o vendedor, o *easypay* fornece entre cem a um milhão de referências de pagamento multibanco que o vendedor deve fornecer ao cliente para efetuar o pagamento. Após o pagamento ser efetuado, o vendedor é notificado pela *easypay* e o valor é transferido para a conta deste.

Na Tabela 3.1 foram analisados serviços de pagamentos eletrónicos que possibilitam pagamentos *MarketPlace* e, como foi concluído, a maior parte desses serviços não funcionam na europa. Porém esses serviços fornecem também sistemas de pagamentos convencionais, C2B, que apresentam menos restrições à localização da sede da empresa que queira utilizar um serviço deste tipo.

3.1.6 Segurança

A segurança tem de estar sempre presente em qualquer transação monetária, e os pagamentos eletrónicos não são exceção. No entanto, existe um vasto leque de casos de fraude envolvendo estes sistemas. Para combater isso, em 2006, as maiores empresas de fornecimento de cartões de crédito (*Visa*, *Master Card*, *American Express*, *Discover* e *JCB International)* juntaram-se e criaram a *Payment Card Industry Data Security Standard* (PCI DSS) [Pcis14]. O PCI DSS é um padrão de boas práticas de segurança no manuseamento de cartões de crédito (ou débito) [Pcis13].

O objetivo deste conjunto de normas é reduzir as fraudes nas transações com recurso a cartões de crédito nos pagamentos eletrónicos. Todavia, para isso ser possível, é necessário que os serviços de pagamento com cartões de crédito apliquem estas normas. Para forçar essa utilização, as empresas que criaram o PCI DSS estabeleceram que o não cumprimento das normas PCI DSS, por parte dos serviços de pagamento, poderia levar a multas pesadas ou à perda da autorização para utilização dos seus cartões nos seus serviços [Logu13].

O padrão PCI DSS apresenta já três versões, datando a mais recente de novembro de 2013. Esta versão conta com 12 requisitos separados em 6 categorias [Pcis13] presentes no Anexo A.

3.2 Dispositivos móveis

O primeiro telemóvel foi apresentado pela Motorola em 1973 em Nova Iorque. O *Motorola Dynatac*, assim era o seu nome, foi inventado por Marin Cooper. Porém, só uma década depois é que a Motorola começou a comercializar o telemóvel, adicionando algumas melhorias como a redução da sua dimensão (33 centímetros de comprimento, 12 de largura e 7 centímetro de espessura) e a redução do seu peso (2,5 quilogramas). Porém o fator preço era desvantajoso, 4000 dólares [Tele00]. Este telemóvel foi o primeiro passo da passagem do telefone convencional para os telefones móveis.

Uma grande evolução nos dispositivos móveis foi o serviço de SMS. Em 1992, Neil Papworth enviou a primeira mensagem SMS da história, através de um computador do escritório

da Vodafone UK para o telemóvel do seu colega Richard Jarvis, com o texto "*Merry Christmas*". O telemóvel que recebeu a primeira mensagem SMS foi o *Orbitel 901*, que também foi um dos primeiros telemóveis a suportar a rede GSM [Gsmh14].

Foi preciso esperar pela década de 90 para surgir o primeiro telemóvel com suporte de aplicações, o *Psion EPOC*. Este telemóvel tinha como sistema operativo o EPOC que permitia a execução de aplicações desenvolvidas em OPL (*open programming language*).

No entanto, a grande evolução das aplicações nos dispositivos móveis deu-se pela criação do sistema operativo *Symbian* que foi adotado por várias marcas de telemóveis, nomeadamente *NOKIA*, *Samsung* e *Sony Ericson*. Este sistema operativo permitia a execução de aplicações desenvolvidas em JAVA ME (J2ME/JME), uma versão do sistema Java da *Sun Microsystems* (adquirida em 2009 pela *Oracle*) com o objetivo de executar aplicações simples em dispositivos com ecrãs pequenos, memória, capacidade gráfica e capacidade de processamento igualmente reduzidas [Orac14].

Os dispositivos com *Symbian* dominaram o mercado até o aparecimento do *smartphone iPhone* em 2007 e posteriormente do sistema operativo *Android*, iniciando-se uma revolução neste mercado.

O *iPhone* foi lançado em 2007 pela empresa norte-americana *Apple* e criou uma revolução nos dispositivos móveis. Este telemóvel, ou melhor dizendo este *smartphone* (telemóvel inteligente) apresentava um ecrã *touchscreen* de 3.5 polegadas com capacidade de múltiplo toque que aumentava a sua usabilidade. Acrescentou um conjunto de sensores de movimento que foram aproveitados para aumentar a qualidade de aplicações como por exemplo, os jogos [Jami11]. Apresentava também um sistema operativo diferente: o *iOS* com uma interface adaptada para o uso do dispositivo sem recuso a teclado, ao contrário do que acontecia até então.

3.2.1 Mercado dos dispositivos móveis

O aparecimento do *Android*, desenvolvido pela *Open Hanset Alliance* e liderado pela *Google*, e igualmente do *iPhone*, levou à mudança do mercado dos dispositivos móveis. Deixam de ser simples dispositivos para efetuar chamadas e enviar SMS e passam a ser um pequeno computador que se pode transportar no bolso.

Estes sistemas operativos, *Android* e *iOS*, trouxeram consigo a massificação das lojas de aplicações. Com estas lojas, a instalação de aplicações num dispositivo móvel tornou-se mais simples, levando ao crescimento da popularidade dos *smartphones*. No final de 2013, 1 em cada 5 pessoas possuía um *smartphone* e 1 em cada 17 possuía também um *tablet* [Hegg13].

Um estudo realizado pela Business Insider [Edwa14], presente na Figura 3.6, demonstra as alterações que este mercado sofreu desde 2009. No estudo é claro o aumento da popularidade do *Android* e do *iOS* desde 2009 até 2014. Em 2009, estes dois sistemas juntos possuíam apenas 20% do mercado, em 2014 juntos têm mais de 90% de cota de mercado. É possível ver a queda dos "outros sistemas", onde está incluído o *Symbian*.

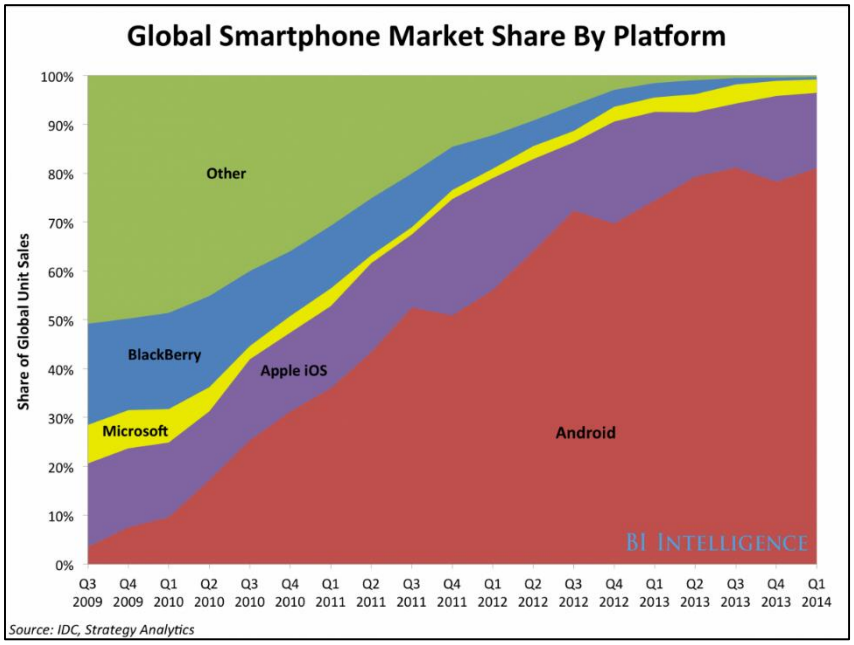

Figura 3.6: Mercado dos dispositivos móveis

3.3 Tecnologias

As tecnologias a serem utilizadas na prova de conceito de uma aplicação para *share economy*, que faz parte desta dissertação, foram escolhidas em colaboração com a empresa de acolhimento e foram também escolhidas tendo em conta que o objetivo é uma prova de conceito e não um produto final.

3.3.1 *Backend*

Aplicações deste tipo obrigam à existência de um *backend* onde a maior parte da lógica de negócio seja implementada. Porém o objetivo desta dissertação foca-se mais no desenvolvimento da própria aplicação de forma a provar o conceito de uma aplicação de entregas. Tendo isto em conta optou-se por utilizar uma tecnologia *noBackend*.

NoBackend

O conceito *noBackend*, apresentado inicialmente por Gregor [Hehr14], é uma forma diferente de desenvolver aplicações que necessitam de um *backend*.

A abordagem tradicional contém um servidor e um conjunto de clientes, no servidor é desenvolvido o *backend* da aplicação e nos clientes é desenvolvido o *frontend* da aplicação. Nesta abordagem, inicialmente é dada mais atenção ao desenvolvimento do modelo de dados do *backend* e também à comunicação que terá de existir entre o servidor e os clientes. Só após essa

componente estar concluída é que se iniciava o desenvolvimento dos clientes, ou seja a componente *frontend*.

Já na abordagem *noBackend* a componente mais importante é o *frontend*, por ser a componente que mais interage com o utilizador. Deste modo, o objetivo é "dissociar as aplicações do *backend*, abstraindo tarefas de *backend* com o código de *frontend*" [Noba14].

Atualmente existe um conjunto de *frameworks* que aplicam o conceito *noBackend*. O *website noBackend.org* [Noba14] apresenta uma lista das *frameworks* que já cumpre os requisitos *noBackend*, referidas no Anexo A.

Muitas dessas *frameworks noBackend* funcionam sobre uma camada de *Backend as a Service* (BAAS), instaladas na *cloud*. Essas mesmas *frameworks* disponibilizam métodos que permitem, por exemplo, criar modelos de dados ou efetuar registos/autenticação de utilizadores com a devida segurança.

A decisão de utilizar uma *framework noBackend* foi tomada tendo em conta o princípio fundamental deste conceito que leva o programador a abstrair-se do *backend* e a focar-se mais na aplicação que está em contacto com utilizador, criando assim uma melhor experiência de utilização ao utilizador da aplicação.

Parse

A *framework noBackend* escolhida foi o *Parse*. Esta implementação de *noBackend* disponibiliza SDKs para várias plataformas, como *Windows 8/8.1*, *iOS*, *Android* ou *JavaScript* que permitem estabelecer uma ligação de uma aplicação ao serviço *cloud* do *Parse*.

O *Parse* permite criar classes de objetos e guardá-los, de uma forma transparente, na *cloud* e localmente, gerindo muito bem a sincronização de dados locais com a *cloud* de uma forma completamente transparente para o programador. O *Parse* também fornece métodos para trabalhar com a localização, esses métodos aumentam a abstração do utilizador sendo mais fácil, por exemplo, acompanhar a localização de um utilizador de uma aplicação.

O *Parse*, desde 2013, faz parte do grupo *Facebook* [Kimm13], sendo assim natural a existência de métodos para fazer ligações ao *Facebook*, permitindo autenticação de utilizadores através da sua conta nessa rede social. Também oferece funções semelhantes para utilizadores do *Twitter*.

O *Parse* conta, também, com um serviço de *Push Notification*, semelhante aos que já são oferecidos pela *Google*, *Apple* e *Microsoft* para os seus respetivos SO móveis. Esses serviços apenas funcionam com o sistema operativo da empresa que o disponibiliza ao contrário do *Parse*, que cobre os vários SO móveis.

O grande problema do *Parse* é o seu preço, embora possa ser utilizado de uma forma gratuita. Gratuitamente o sistema fica limitado a apenas 30 pedidos por segundo e apenas uma tarefa a ser executada de cada vez (sem concorrência). O melhor plano oferece 400 pedidos por segundo e 20 *background jobs* em simultâneo mas, com o custo de 3700 € por mês. O *Parse* permite desenvolver código de *backend*, porém a sua execução é também limitada. É possível criar o

chamado *cloud cod*e que são funções que podem ser chamadas pelos clientes, estas funções estão limitadas a um tempo de execução de 15 segundos. É possível também executar *triggers* antes e depois de um objeto ser guardado. Estes *triggers* têm também um tempo limite de execução de 3 segundos. Outra funcionalidade do *Parse* com limitações é a utilização de *background jobs* que na versão gratuita apenas pode ser executado um de cada vez e estando limitados a um tempo de execução máximo de 15 minutos [Pars15a].

Apesar destas limitações, as mais-valias que já foram apresentadas anteriormente permitem que o desenvolvimento de um protótipo de uma aplicação seja mais rápido do que se utilizasse uma abordagem mais traicional. Tendo em conta isso e juntando a facilidade de interligar o *Parse* a uma aplicação *iOS*, o *Parse* foi a tecnologia escolhida para o desenvolvimento do *backend*.

3.3.2 Aplicação

Para o desenvolvimento da aplicação existiam várias possibilidades, podendo escolher-se um de três grandes tipos de aplicações: nativas, hibridas ou aplicação *web* [Budi13].

As aplicações nativas são aplicações que se podem instalar a partir das lojas dos SO móveis e que ficam disponíveis para o utilizador através de um ícone no ecrã do dispositivo. Estas aplicações conseguem facilmente obter dados dos vários sensores do dispositivo.

Já as aplicações *web* são páginas *web* otimizadas para os dispositivos móveis. Desenvolvidas em *HTML5*, *JavaScript e CSS3*, em geral são otimizações (através de redirecionamento) de aplicações *web* já existente para outros tipos de computadores. Podem também ser instaladas como aplicações nativas que apresentam o conteúdo das páginas *web*.

As aplicações hibridas são também, em alguns casos, desenvolvidas em *HTML5, JavaScript* e *CSS3* mas têm a possibilidade de aceder a vários sensores ou outros componentes que o dispositivo móvel contém, como por exemplo a câmara ou o acelerómetro. As aplicações hibridas são desenvolvidas através de *frameworks* como o *PhoneGap* (também designado como *Apache Cordova*) que permitem criar uma aplicação que funcione em vários SO móveis [Bris15].

A nível de performance as aplicações nativas demonstram melhores resultados, e tendo em conta este ponto e a experiência da empresa de acolhimento, a escolha recaiu neste tipo.

Porém dentro das aplicações nativas é necessário realizar outra escolha, o SO móvel alvo. Na secção 3.2.1 é apresentado um estudo em que se indica a cota de mercado dos utilizadores de cada SO móvel, onde o SO *Android* é claramente dominador seguindo-se o SO *iOS*. A escolha do sistema operativo alvo poderia ter sido realizada só tendo em conta este parâmetro porém foram considerados também os seguintes:

- O valor gasto pelos utilizadores destes sistemas nas lojas de aplicações dos respetivos sistemas operativos;
- Exemplos de aplicações semelhantes;
- Interesse do estudante em aprender tecnologias diferentes.

Um estudo realizado pela *Online Publishers Association* [Rich12], em 2012, analisou o valor monetário gasto pelos utilizadores em compras nas lojas de aplicações do *Android* e de *iOS*. O estudo concluiu que 66% dos utilizadores de *Android* inquiridos não efetua qualquer compra de aplicações na loja da *Google*, e apenas 34% dos mesmos admitiu gastar algum dinheiro em compras de aplicações. Todavia, no caso dos utilizadores de *iOS*, os resultados apresentam uma tendência contrária aos utilizadores de *Android*. Apenas 30% dos utilizadores de *iOS* inquiridos admitiu não gastar nenhum dinheiro na compra de aplicações na loja da *Apple*, menos 24% que para o *Android*. O estudo conclui assim que os utilizadores de *iOS* estão mais dispostos a efetuar pagamentos para utilizarem aplicações. A Figura 3.7 apresenta mais detalhadamente os dados recolhidos com o estudo.

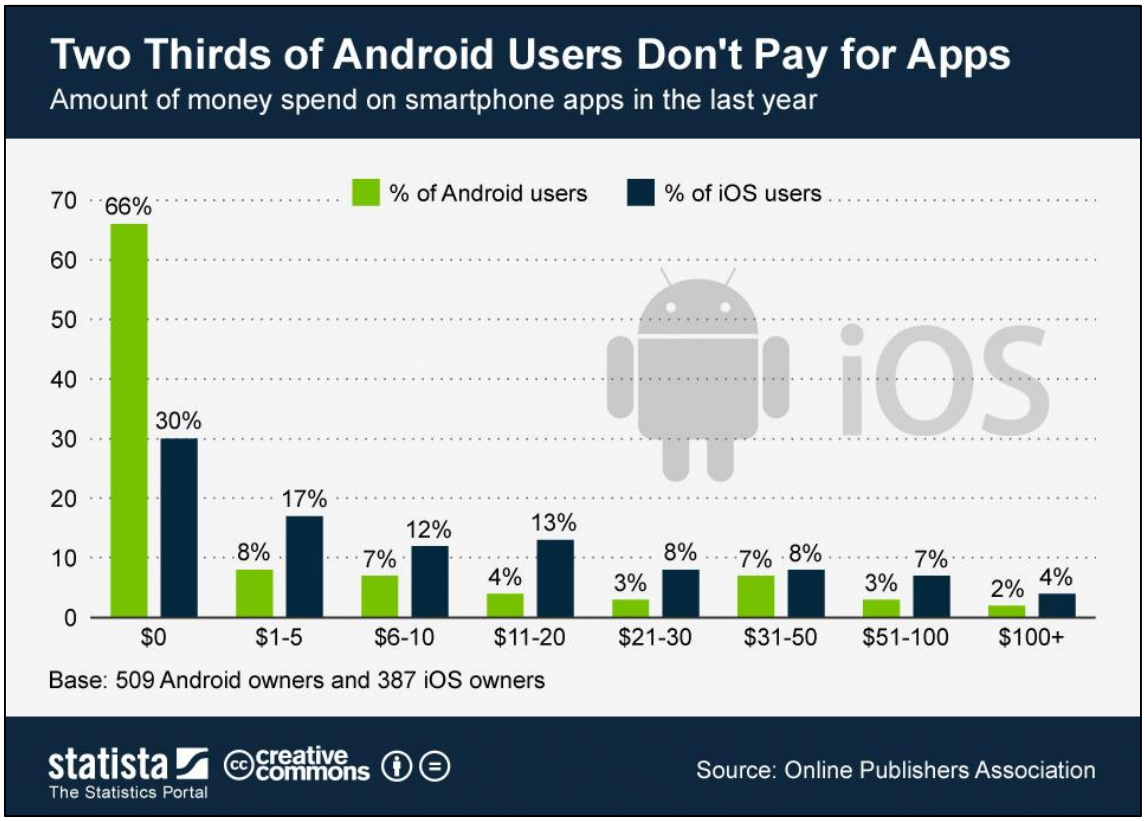

Figura 3.7: Dinheiro gasto em lojas de aplicações durante um ano

O outro paramento da escolha consistiu em analisar as aplicações *share economy* que já estão no mercado. Após serem analisadas várias aplicações chegou-se à conclusão que muitas dessas aplicações têm apenas versões *web* ou para *iOS*. Em alguns casos, havia também versões para *Android*. Também foi possível perceber que as empresas disponibilizavam em primeiro lugar aplicações para *iOS* e só numa fase posterior é que as portavam para *Android*, ou outros sistemas operativos móveis.

O último ponto, mas não menos importante, foi o interesse do estudante em aprender algo diferente do que tinha feito até então. E tendo em conta estes pontos a escolha recaiu por desenvolver uma aplicação nativa para *iOS*.

iOS

Este sistema operativo móvel foi desenvolvido pela *Apple*, inicialmente para ser utilizado no *iPhone* e no *iPod Touch*. Mais tarde, a sua utilização, foi estendida para *iPad* e para a *Apple TV*. O *iOS* foi desenvolvido tendo como base o *OSX* que por sua vez teve como base o UNIX.

O *iOS* é uma sistema desenvolvido em camadas [Appl14], como está descrito na Figura 3.8.

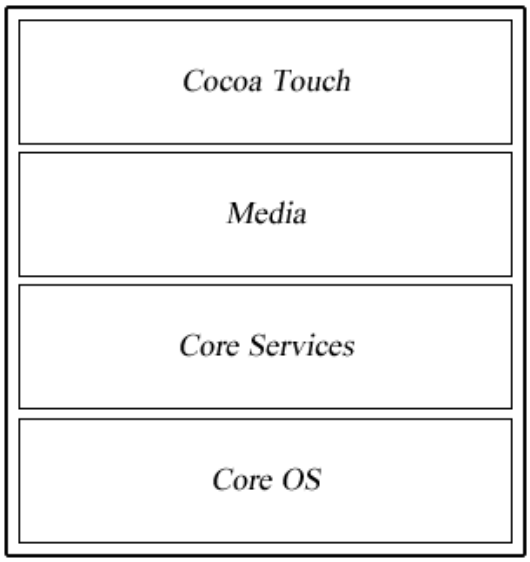

Figura 3.8: Camadas do *iOS*

A *Apple* disponibiliza *frameworks* para utilizar qualquer uma das camadas porém, é mais aconselhável trabalhar com as classes de mais alto nível.

A camada *Core OS* é a camada mais baixa e é utilizada pelas restantes camadas. Os programadores só devem recorrer a esta camada quando precisam de trabalhar a mais baixo nível como por exemplo, em questões de segurança.

A camada *Core Services* disponibiliza as *frameworks* que contêm os tipos de dados mais utilizados tais como, as *frameworks Foundation*. Contêm também serviços ligados à localização, à rede e também ao serviço de *cloud* da *Apple iCloud*.

A camada *Media* contém as tecnologias ligadas à multimédia. Contém tecnologias para computação gráfica, como *OpenGL*; tecnologias ligadas ao áudio, como *OpenAL*; vídeo e *AirPlay* que é um serviço que permite a ligação à *Apple TV*.

Por fim existe a camada *Cocoa Touch*, que é a camada de mais alto nível do *iOS*. Esta camada contém as *frameworks* chave para o desenvolvimento de uma aplicação, como a aparência base de uma aplicação, a interação com o toque, ligação ao serviço de *push notification* e acesso a *frameworks* de alto nível ligadas à multimédia.

O *iOS*, como já foi referido, é o sistema operativo de quatro dispositivos diferentes:

- ***iPod Touch***: é um reprodutor de multimédia e um assistente pessoal digital (PDA) [Nune11]. Apresenta capacidade de múltiplo toque e internet sem fios, tendo as gerações mais recentes uma câmara que permite a captura de fotografias e a gravação de vídeo.

Existem já várias gerações deste dispositivo sendo a atual, a quinta versão, disponibilizada ao público em 2012.

- ***iPhone***: é um *smartphone* que agrupa as funcionalidades que estão disponíveis no *iPod Touch* acrescentando as capacidades telefónicas e de envio de mensagens de texto e *Visual Voicemail* [Nune11]. Apresenta também várias versões sendo a mais recente, o *iPhone 6* e o *iPhone 6 Plus*.

- ***iPad***: é o *tablet* da *Apple*. Este dispositivo, lançado em 2010, encontra-se a meio caminho entre o *iPhone* e um computador pessoal e fornece capacidades semelhantes ao *iPhone*. Porém, apresenta um ecrã de maior dimensão levando a um aumento de produtividade em alguns casos. O *iPad* apresenta várias versões estando neste momento na quarta geração – *iPad Air 2*. Existe uma outra versão do *iPad*, o *iPad mini*, que apresenta um ecrã com menores dimensões.

- ***Apple TV***: é o centro de multimédia da *Apple* que funciona ligando o dispositivo a uma televisão. Tem a capacidade de reproduzir filmes e músicas.

Após a escolha do SO móvel restava escolher a linguagem em que a aplicação seria desenvolvida. Para desenvolver aplicações nativas é possível desenvolver em *Objective-C*, uma variante da linguagem C, ou em *Swift*, uma linguagem recente (apresentada em junho de 2014). Neste caso a empresa de acolhimento aconselhou que a aplicação fosse desenvolvida em *Objective-C* por dois motivos: como o *Swift* é recente existe um maior número de conteúdos que ajudam na aprendizagem do *Objective-C*; por outro lado a empresa considera que o *Swift*, devido ao seu tempo de vida, pode apresentar algumas lacunas no momento da implementação da aplicação, sendo desse modo preferível desenvolver em *Objective-C*.

Objective-c

O *Objective-c* é uma linguagem orientada a objetos, que define extensões à linguagem *C*. Ao contrário de algumas linguagens derivadas do *C*, em *Objective-c* é possível compilar ficheiros desenvolvidos em C, pois é apenas mais uma camada adicionada ao *C standard* [Buca13].

O *Objective-c* foi desenvolvido por Brad Cox e Tom Love, que aproveitaram a sintaxe da linguagem *Smalltalk* (uma das primeiras linguagens orientadas a objetos) para adicionar ao *C* suporte para objetos, processando essas construções num pré-processador de forma a manter a compatibilidade com a linguagem C [Jack11, Nune11]. A sintaxe do *Objective-C* permite fazer especificações, à base de mensagens (Figura 3.9), que manipulam objetos.

Em 1988, o *Objective-c* foi licenciado pela *NeXT*, uma empresa criada por Steve Jobs após o seu afastamento da *Apple* e, desta forma, lançou um compilador próprio com bibliotecas próprias para construção de interfaces de utilizador. Em 1996, a *NeXT* foi adquirida pela *Apple* e o *Objective-c* foi utilizado para contruir grande parte do *Mac OS X*. Atualmente, a biblioteca de

interface de utilizador é a *Cocoa* que teve como base as bibliotecas criadas pela *NeXT* [Jack11]. A *Apple* apresentou também um IDE, o *Project Builder*, que permitia a construção dessas mesmas interfaces e o desenvolvimento de código *Objective*-C. Esse IDE foi mais tarde substituído pelo atual *XCode*.

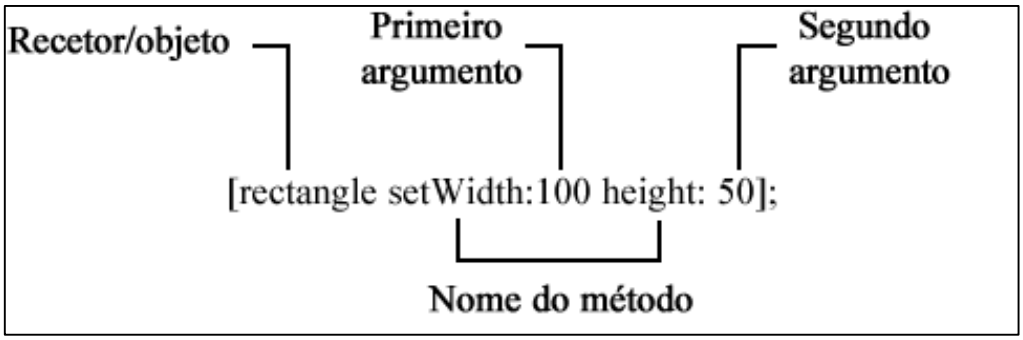

Figura 3.9: Mensagens *Objective-c*

3.3.3 Pagamentos eletrónicos.

Na secção 3.1 foi realizada uma análise detalhada aos pagamentos eletrónicos. Nessa análise foi dado um foco especial aos serviços de pagamento que suportam *MarketPlace*. No entanto, como foi referido nessa secção, esses serviços embora sejam os sistemas de pagamentos mais indicados para aplicações *share economy* apresentam um conjunto de problemas, nomeadamente as restrições à localização da sede da empresa que queira utilizar o serviço e o seu custo.

Esse problema foi discutido com a empresa de acolhimento, e chegou-se a conclusão que para efeitos de prova de conceito poderia ser utilizado um serviço de pagamentos C2B, em que os pagamentos seriam realizados do cliente para a empresa, e mais tarde (semanalmente ou até mensalmente) eram realizados os pagamentos aos estafetas. Este processo é semelhante ao que outras empresas com serviços *share economy* utilizam, como é o exemplo da *Uber.*

Existem muitos serviços que permitem utilizar pagamentos C2B, mas neste caso não foi feita uma análise detalhada a todos de forma a escolher a melhor solução. Por sugestão da empresa de acolhimento optou-se por utilizar um serviço recente, que pertence a uma *spin-off* da empresa, o *Switch Payments*.

Switch Payments

O serviço foi lançado em 2015, e encontra-se ainda numa fase inicial mas apresenta uma API para programadores bastante simplificada, sendo muito simples integrar um sistema de pagamentos numa aplicação móvel ou numa página *web* [Paym15].

Aceita pagamentos de qualquer parte do mundo em mais de 150 tipos de moedas diferentes. Ao nível da segurança o *Switch Payments* compre com os padrões de segurança PCI nível 2.

Na secção 3.1.3 é descrito como é processado um cartão de crédito, e é apresentado um conjunto de entidades responsáveis por o processar. O *Switch Payments* entra nesse processo para ser uma ponte entre o *Comerciante* e as restantes entidades do processo, retirando do *Comerciante* algumas preocupações de segurança que existem no processamento de cartões de crédito e abstraindo o processo. A utilização do *Switch Payments* evita que dados sensíveis do cartão passem pelo *Comerciante* sendo enviados diretamente para o *Switch Payments*. A Figura 3.10 apresenta os fluxos de comunicação existente no processamento de pagamentos com recurso a cartões de crédito.

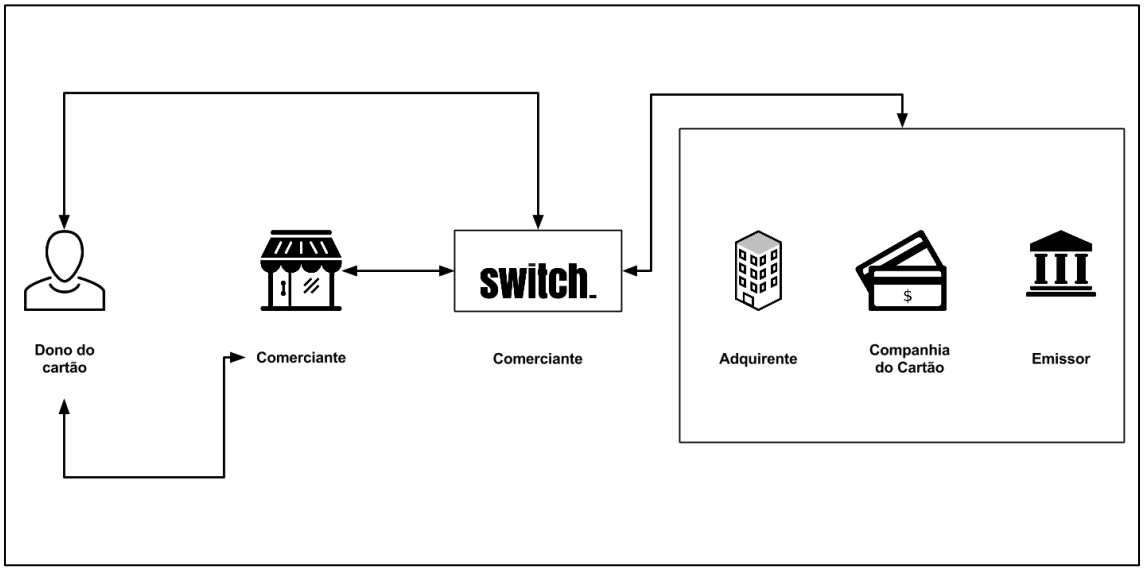

Figura 3.10: *Switch Payments*

Capítulo 4

Requisitos e Arquitetura

Neste capítulo serão abordados os requisitos da aplicação de prova de conceito *share economy* e de seguida a conceção da mesma. Focam-se primeiramente os requisitos iniciais da aplicação, transformando os mesmos em narrativas de utilização, que são apresentados com uma classificação quantitativa em *story points* referentes ao esforço de implementação e uma classificação qualitativa referente à prioridade dessa narrativa de utilização.

Como alguns requisitos e narrativas de utilização são apresentados como problemas de implementação, estes são analisados e são apresentadas funcionalidades da aplicação e do serviço que possam resolver esses problemas.

Após algumas ideias de resolução, é apresentada a arquitetura de todo o sistema, começando por visão de alto nível. Finalmente especifica-se o modelo de dados a implementar no *backend*.

4.1 Requisitos

4.1.1 Objetivos

O principal objetivo passa pela criação de uma aplicação móvel que permita a um cliente encomendar refeições de um restaurante, apresentado na aplicação, e acompanhar o estado da mesma, nomeadamente perceber se já tem um estafeta atribuído à sua encomenda, se o estafeta já recolheu a refeição ou se este já está a dirigir-se para sua residência.

Por outro lado, a aplicação também poderá ser utilizada por um estafeta que deverá receber os pedidos de encomendas, que pode aceitar ou recusar. A aplicação deve permitir ao estafeta atualizar o estado da entrega para que o cliente o vá percebendo ao longo do tempo.

É também importante que a aplicação criada esteja preparada para lidar com conflitos de utilizadores, recolhendo dados importantes para tomar parte nesses conflitos.

De forma a cumprir estes objetivos foi elaborada uma lista de requisitos que é descrita na próxima secção.

4.1.2 Levantamento de Requisitos

Na fase inicial foi criada uma lista de requisitos que a aplicação e o serviço inerente à aplicação teriam de cumprir no final desta dissertação. Esses requisitos foram classificados com uma prioridade (alta, média ou baixa). Sendo os requisitos de prioridade alta fundamentais para demonstrar a prova de conceito, e os de baixa prioridade aqueles que não estão inteiramente ligados com o fluxo principal dessa prova de conceito.

Juntamente com os requisitos foi também definido um conjunto de atores do sistema: o *utilizador não autenticado*, o *cliente*, o *estafeta* e o *gestor de sistema*, visíveis na Figura 4.1.

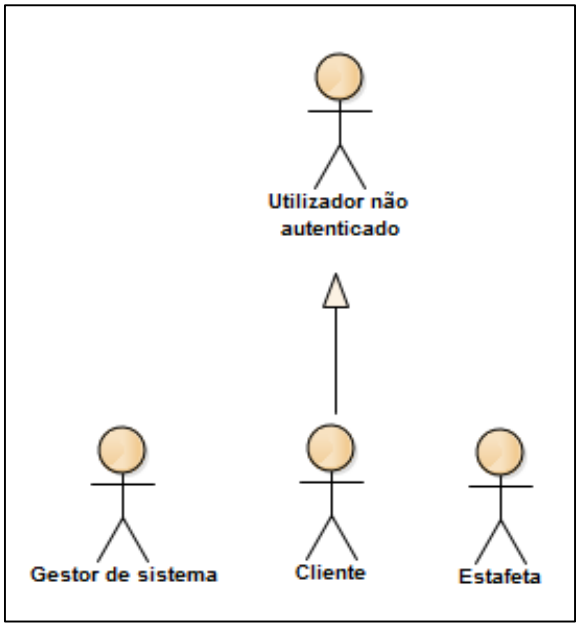

Figura 4.1: Atores

Um *utilizador não autenticado* é um utilizador que ainda não se registou ou não iniciou sessão na aplicação. Um *cliente* é um utilizador que iniciou sessão e anteriormente se registou como utilizador. Será esta entidade que poderá criar uma encomenda. Um *estafeta* é um utilizador que iniciou sessão e anteriormente foi registado como *estafeta*.

A Tabela 4.1 apresenta esses mesmos requisitos com a respetiva prioridade.

Tabela 4.1: Requisitos

Código	***Requisito***	***Prioridade***
UNA01	A aplicação deve permitir um *utilizador não autenticado* registar-se como *cliente*.	Alta
UNA02	A aplicação deve permitir um *utilizador não autenticado* iniciar sessão.	Alta
CLI01	A aplicação deve permitir que um *cliente* ou *utilizador não autenticado* possa pesquisar restaurantes utilizando vários critérios.	Média

CLI02	A aplicação deve permitir que um *cliente* ou *utilizador não autenticado* possa consultar a informação de um restaurante (localização, menus e seus preços).	Alta
CLI03	A aplicação deve permitir que o *cliente* ou *utilizador não autenticado* adicione menus ao seu carrinho de encomenda.	Alta
CLI04	A aplicação deve permitir que o *cliente* finalize a encomenda, ficando à espera que lhe seja atribuído um *estafeta*.	Alta
CLI05	A aplicação deve permitir que um *cliente* associe o seu cartão de crédito para efetuar os pagamentos do serviço aos *estafetas*	Alta
CLI06	A aplicação deve permitir que um *cliente* possa acompanhar o estado da encomenda.	Alta
CLI07	A aplicação deve permitir que um *cliente* possa avaliar o *estafeta* após receber a sua encomenda.	Alta
CLI08	A aplicação deve permitir que um *cliente* possa reportar problemas com a sua encomenda.	Alta
CLI09	A aplicação deve permitir que um *cliente* possa confirmar que a entrega foi realizada nas melhores condições.	Alta
CLI10	A aplicação deve apresentar ao *cliente* uma previsão da hora de entrega.	Média
CLI11	A aplicação deve permitir que um *cliente* possa avaliar o restaurante após receber a sua encomenda.	Média
EST02	A aplicação deve permitir que o *estafeta* possa entrar em modo de espera de pedidos.	Alta
EST03	A aplicação deve notificar o *estafeta* que recebeu um pedido, quando este se encontrar em modo de espera.	Alta
EST04	A aplicação deve permitir ao *estafeta* aceitar ou recusar um pedido.	Alta
EST05	A aplicação deve permitir ao *estafeta* consultar informações do restaurante do pedido antes de o aceitar.	Alta
EST06	A aplicação deve permitir ao *estafeta* consultar informações do *cliente* antes de aceitar a encomenda.	Alta
EST07	A aplicação deve permitir que o *estafeta* registe o valor da encomenda no sistema, juntando uma prova desse valor.	Alta
EST08	A aplicação deve permitir que o *estafeta* altere o estado da encomenda.	Alta
EST09	A aplicação deve permitir que o *estafeta* avalie o cliente após a realização da encomenda.	Alta
EST10	A aplicação deve permitir que o *estafeta* indique que a encomenda foi entregue.	Alta

EST11	A aplicação deve permitir que o *estafeta* reporte problemas na entrega da encomenda.	Alta
APP01	A aplicação deve criar um canal de comunicação entre o *cliente* e o *estafeta* durante a realização da entrega.	Média
APP02	A aplicação deve guardar o percurso realizado do *estafeta*.	Alta
APP03	A aplicação deve garantir que as transações monetárias são realizadas em segurança.	Alta
APP04	A aplicação deve estar preparada para ajudar um *gestor de sistema* a resolver um conflito entre utilizadores.	Alta
BO01	O *back office* deve permitir que um *gestor do sistema* possa ver, adicionar e editar restaurantes.	Alta
BO02	O *back office* deve permitir que um *gestor do sistema* possa ver, adicionar e editar menus de restaurantes.	Alta
BO03	O *back office* deve permitir que um *gestor do sistema* possa consultar informação dos *estafetas* e validar e bloquear *estafetas*.	Alta
BO04	O *back office* deve permitir que um *gestor do sistema* possa consultar as entregas realizadas.	Alta
BO05	O *back office* deve permitir que um *gestor de sistema* possa consultar os problemas reportados pelos *clientes* e *estafetas*.	Alta
BO06	O *back office* deve permitir que um *gestor do sistema* possa consultar a informação dos *clientes*.	Alta
BO07	O *back office* deve permitir que um *gesto do sistema* possa adicionar novos estafetas.	Alta

O requisito **UNA01** está relacionado com o registo por parte de um utilizador na aplicação. O *utilizador não autenticado* terá de fornecer o seu nome, endereço de correio eletrónico, número de contacto e uma palavra passe de forma a autenticar-se no serviço. Mais tarde deverá adicionar um cartão de crédito como está descrito no requisito **CLI05.** Um utilizador *não autenticado* não se pode registar como *estafeta*. O registo como *estafeta* só poderá ser realizada após a pessoa ser entrevistada pela equipa de *gestores da aplicação*. Após a entrevista, o *gestor do sistema* pode registar o *estafeta* no sistema (requisito **BO07**).

Quando um *cliente* finaliza uma encomenda (requisito **CLI04**), fica à espera que lhe seja atribuído um *estafeta*.

O *estafeta*, quando estiver no restaurante, deve confirmar se o preço do menu é o indicado pela aplicação e, caso não seja, deve notificar o cliente dessa alteração.

A aplicação deve ajudar a resolver um conflito entre o *cliente* e o *estafeta*, apresentando mecanismos que levem a perceber o que realmente aconteceu e quem tem a razão nesse conflito. Os requisitos **APP02** e **APP04** surgem com esse objetivo.

Por outro lado, a aplicação terá de permitir validar a entrega por parte do *estafeta* e por parte do *cliente* (requisitos **EST10 e CLI08**).

A aplicação terá também um sistema de classificações (requisitos **EST09** e **CLI07**) semelhante ao sistema utilizado pelo *eBay*, já analisado na secção 2.3.1, de forma a permitir que os utilizadores se possam avaliar mutuamente após uma transação entre eles.

A aplicação será completada com um *back office*. Contará com funcionalidades de gestão do sistema, nomeadamente a gestão de restaurantes (**BO01**), a gestão de menus dos restaurantes (**BO02**), validar e bloquear *estafetas* (**BO03**), visualizar os dados das entregas realizadas (**BO04**), consultar informações de *estafetas* e *clientes* (**BO03 e BO06**) e gerir os problemas reportados pelos *clientes* e *estafetas*. Esta gestão é da responsabilidade dos *gestores de sistema*.

Os requisitos apresentados levaram a que fossem pensadas funcionalidades importantes que a aplicação tivesse de apresentar.

4.2 Narrativas de utilização

Após serem definidos os requisitos foram criados um conjunto de narrativas de utilização. Essas narrativas de utilização foram estimadas em *story points* seguindo a seguinte escala: 0, 1, 2, 3, 5, 8, 13, 20, 40, 100. E, da mesma forma que os requisitos, as narrativas de utilização também foram classificados com uma prioridade (alta, média ou baixa), que segue o mesmo critério utilizado nos requisitos.

A Tabela 4.2 apresenta as narrativas de utilização criadas, utilizando o mesmo conjunto de atores referido nos requisitos. Esta tabela não apresenta as narrativas de utilização da componente *back office* já que o seu desenvolvimento não está presente nos objetivos desta dissertação.

Tabela 4.2: Narrativas de utilização

Código	***Descrição***	***Prioridade***	***Story Points***
UNA01	Como *utilizador não autenticado*, quero criar uma conta na aplicação.	Média	8
UNA02	Como *utilizador não autenticado*, quero criar uma conta na aplicação utilizando a minha conta do *Facebook*.	Baixa	5
UNA03	Como *utilizador não autenticado*, quero iniciar sessão na aplicação utilizando o *Facebook*.	Baixa	8
UNA04	Como *utilizador não autenticado*, quero iniciar sessão na aplicação utilizando uma conta criada anteriormente.	Média	8
UNA05	Como *utilizador não autenticado*, quero utilizar a aplicação sem iniciar sessão.	Alta	3
UNA06	Como *utilizador não autenticado*, quero ver os restaurantes perto de uma determinada localização.	Alta	13
UNA07	Como *utilizador não autenticado*, quero alterar a localização de listagem de restaurantes.	Baixa	5
UNA08	Como *utilizador não autenticado*, quero ter uma previsão do tempo da entrega antes de escolher um restaurante.	Média	8
UNA09	Como *utilizador não autenticado*, quando estou a ver as informações de um restaurante quero ver os produtos	Média	5

	organizados por categorias.		
UNA10	Como *utilizador não autenticado*, quero escolher um produto, escolher a sua quantidade e ver o preço total.	Alta	8
UNA11	Como *utilizador não autenticado*, ao escolher um produto, quero ter a possibilidade de adicionar observações para serem lidas pelo estafeta.	Média	5
UNA12	Como *utilizador não autenticado*, quero adicionar um produto ao carrinho.	Alta	5
UNA13	Como *utilizador não autenticado*, depois de adicionar um produto ao carrinho quero adicionar mais produtos.	Alta	5
UNA14	Como *utilizador não autenticado*, quero editar os itens do carrinho.	Baixa	8
UNA15	Como *utilizador não autenticado*, quero fazer *checkout* da encomenda.	Alta	20
UNA16	Como *utilizador não autenticado*, quero iniciar sessão ou registar-me antes de fazer *checkout*.	Média	8
CLI01	Como *cliente*, quero adicionar informações adicionais (se ainda não o tiver feito), tais como o cartão de crédito, antes de realizar o *checkout*.	Média	5
CLI02	Como *cliente*, quero confirmar a morada de entrega antes de realizar o *checkout*.	Média	5
CLI03	Como *cliente*, quero confirmar a encomenda antes de realizar o *checkout*.	Alta	5
CLI04	Como *cliente*, quero acompanhar o estado da encomenda.	Alta	5
CLI05	Como *cliente*, quero ser notificado caso existam alterações de preço dos produtos, e aceitar ou recusar essas alterações.	Média	8
CLI06	Como *cliente*, quero ter uma previsão do tempo que resta até a encomenda ser entregue.	Baixa	13
CLI07	Como *cliente*, quero comunicar com o estafeta durante a realização da entrega.	Baixa	20
CLI08	Como *cliente*, quero ter o código de confirmação visível, durante a realização da entrega.	Média	3
CLI09	Como *cliente*, quero avaliar o estafeta após a entrega.	Média	8
CLI10	Como *cliente,* quero avaliar o restaurante após a encomenda ser entregue.	Media	5
CLI11	Como *cliente*, quero relatar eventuais problemas que existam durante a entrega.	Média	13
CLI12	Como *cliente*, quero relatar eventuais problemas que existam com a refeição.	Baixa	8
CLI13	Como *cliente*, quero relatar eventuais problemas que existam com a aplicação.	Baixa	5
CLI14	Como *cliente*, quero consultar as encomendas que já realizei.	Baixa	8
CLI15	Como *cliente*, quero consultar o meu perfil.	Baixa	5
CLI16	Como *cliente*, quero editar o meu perfil.	Baixa	8
CLI17	Como *cliente,* quero poder pedir uma fatura do serviço pago pela entrega da encomenda.	Baixa	5
EST01	Como *estafeta*, quero iniciar o modo de “espera de pedidos”	Alta	5
EST02	Como *estafeta*, ao receber um pedido, quero ver as informações	Média	8

	gerais da encomenda.		
EST03	Como *estafeta*, ao receber um pedido, quero ver as informações detalhadas do restaurante, *cliente* e encomenda antes de aceitar.	Média	8
EST04	Como *estafeta*, ao receber um pedido, quero ver as informações do restaurante.	Média	8
EST05	Como *estafeta*, ao receber um pedido, quero poder aceitar ou recusar o pedido.	Alta	8
EST06	Como *estafeta*, quero poder confirmar a encomenda, ligando para o restaurante, antes de me dirigir ao mesmo.	Média	5
EST07	Como *estafeta*, ao chegar ao restaurante, quero confirmar a encomenda.	Média	8
EST08	Como *estafeta*, ao chegar ao restaurante, quero editar a encomenda.	Baixa	13
EST09	Como *estafeta*, após a confirmação da encomenda, quero ver o NIF da empresa para a fatura ser emitida.	Baixa	3
EST10	Como *estafeta*, após o pagamento, quero fotografar a fatura para provar a compra.	Média	20
EST11	Como *estafeta*, após fotografar a fatura, quero ver as informações do *cliente*.	Média	8
EST12	Como *estafeta*, ao chegar à localização do *cliente*, quero confirmar a entrega.	Alta	13
EST13	Como *estafeta*, ao chegar à localização do *cliente*, se não o conseguir encontrar, quero enviar uma fotografia da minha localização.	Média	20
EST14	Como *estafeta*, após a entrega, quero avaliar o *cliente*.	Média	8
EST15	Como *estafeta*, durante a entrega, quero contactar com o *cliente*.	Baixa	20
EST16	Como *estafeta*, quero consultar o meu histórico de entregas.	Baixa	8
EST17	Como *estafeta*, quero ver o meu perfil.	Baixa	5
EST18	Como *estafeta*, quero editar o meu perfil.	Baixa	8

As narrativas de utilização presentes na Tabela 4.2 são referentes a todo o sistema e são apresentadas de uma forma bastante detalhada. De forma a simplificar, foi criado um conjunto de diagramas de casos de uso que apresentam as narrativas de utilização mais relevantes com uma escrita mais simplificada e agrupando algumas das narrativas de utilização.

A Figura 4.2 apresenta os casos de uso mais relevantes para o fluxo principal da prova de conceito da aplicação em modo de *cliente*. É de salientar que um *utilizador não autenticado* tem a possibilidade de utilizar grande parte da aplicação sem realizar a autenticação. Essa autenticação só será solicitada se o mesmo necessitar de realizar uma encomenda. Um *Cliente* possui os mesmos casos de uso do *utilizador não autenticado* com a exceção dos casos de uso relacionados com o registo e início de sessão.

À semelhança da Figura 4.2, a Figura 4.3 apresenta os casos de uso mais relevantes para o fluxo da prova de conceito em modo de *estafeta*. Além dos casos de uso apresentados no diagrama, para um utilizador poder utilizar a aplicação em modo *estafeta* terá de primeiro iniciar sessão, como um *utilizador não autenticado*, utilizando as credenciais de um *estafeta*.

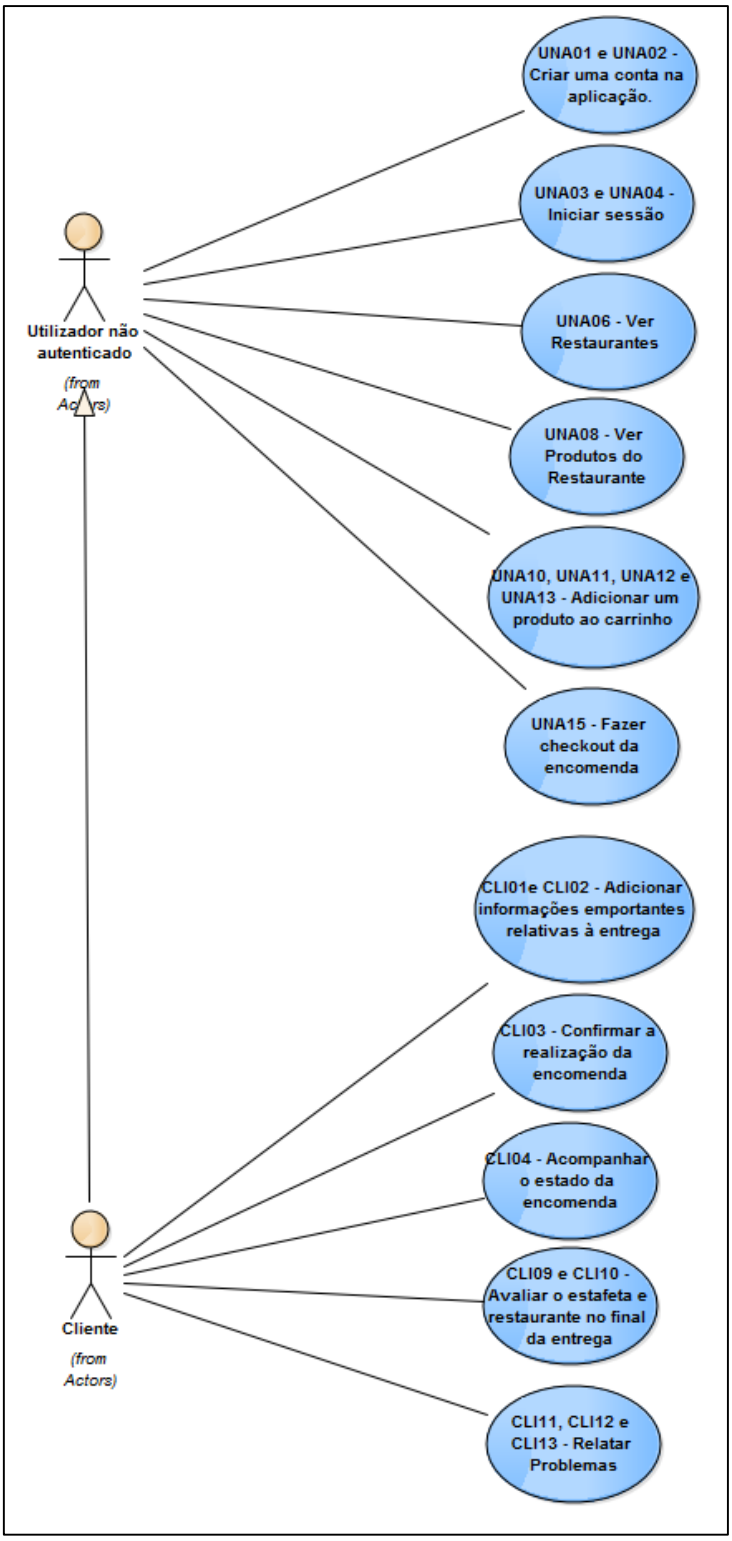

Figura 4.2: Casos de uso da aplicação em *modo Cliente*

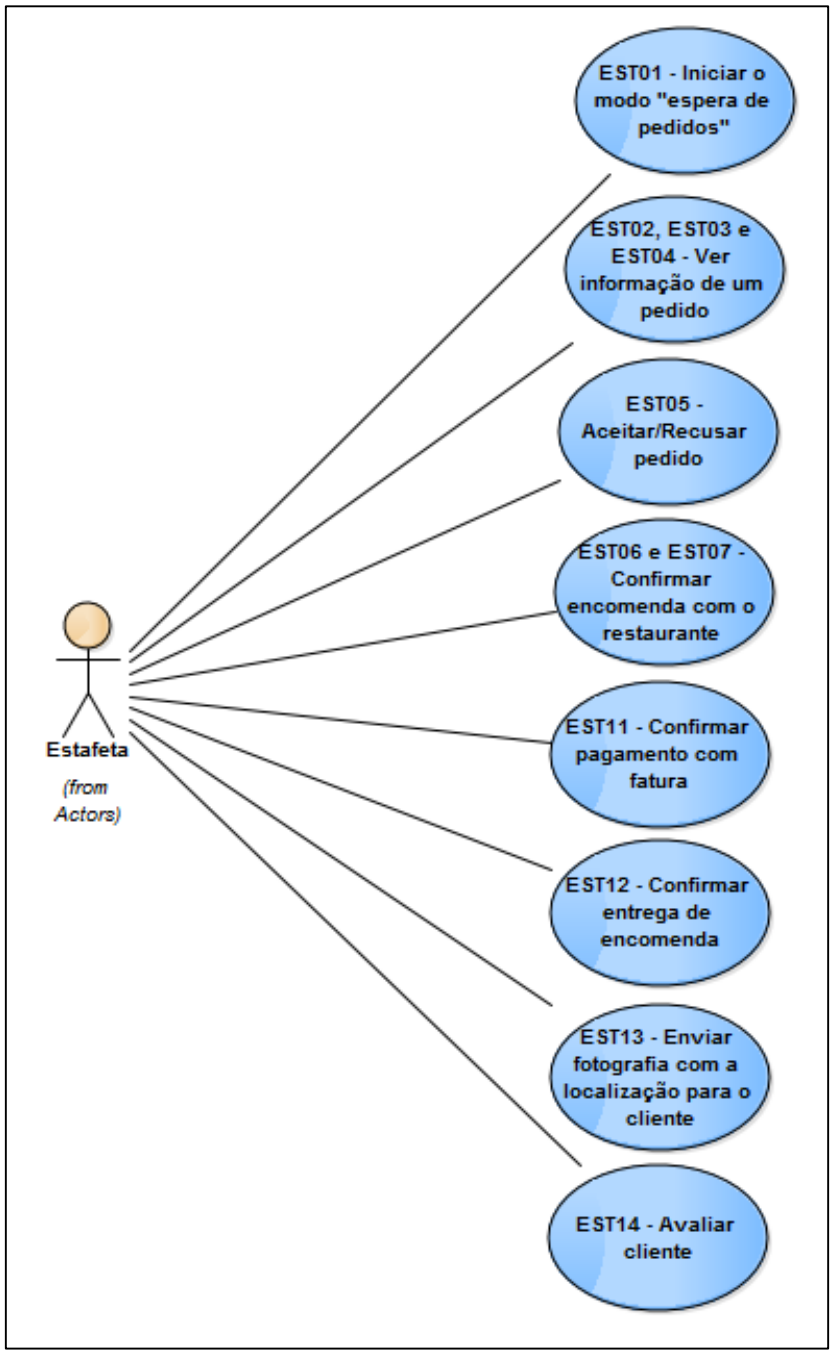

Figura 4.3: Casos de uso da aplicação em *modo Estafeta*

4.3 Funcionalidades

Algumas das narrativas de utilização e requisitos já apresentados carecem de solução óbvia, sendo problemas para os quais teve de ser concebida uma ideia de solução. Dessa forma serão abordados esses problemas explicando as funcionalidades que foram implementadas na aplicação e no serviço de forma a resolvê-los.

Os requisitos **CLI09** e **EST10** estão relacionados com a validação de entrega de uma encomenda. A solução encontrada para este problema passou pela atribuição de um PIN de confirmação de entrega que é fornecido ao cliente, por cada encomenda que este criar. Esse PIN deverá ser depois introduzido no *smartphone* do estafeta pelo próprio cliente de forma a confirmar que recebeu a encomenda.

O conjunto de requisitos **APP02** e **APP04** estão relacionados com a resolução de conflitos que possam acontecer entre utilizadores. Uma das formas passa por guardar o percurso percorrido pelo estafeta (**APP02**), de modo a perceber se os estafetas se deslocaram ao restaurante correto, por exemplo. Porém, esta funcionalidade não cobre todos os tipos de conflitos entre utilizadores que podem acontecer. Um outro ponto crítico, ao nível dos conflitos, acontece no momento da entrega. Neste momento podem acontecer várias situações:

- O estafeta não encontra o cliente
- O estafeta não faz deliberadamente a entrega ao cliente
- O cliente não encontra o estafeta

A solução encontrada para cobrir estas três situações passa por implementar outras três funcionalidades. Quando um estafeta não encontra o cliente pode enviar uma fotografia, que demonstre a sua localização. Ao enviar a fotografia, o estafeta deve permanecer no local durante cinco minutos esperando pelo cliente. Se este não aparecer o estafeta poderá abandonar o local. No entanto, de forma a prevenir que o estafeta abandone o local antes de completar os cinco minutos, a localização do estafeta é monitorizada e este é notificado caso abandone o local. Por outro lado, após os cinco minutos decorrerem e o estafeta der como cancelada a encomenda, o cliente recebe uma notificação à qual pode responder. A sua resposta será analisada juntamente com os dados recolhidos do estafeta e será decidido se o cliente deve fazer o pagamento da encomenda ou não. Por último, quando o estafeta apresenta atraso na sua entrega o cliente poderá entrar em contacto com os gestores de sistema, que tentarão perceber o que se passa com a entrega da encomenda.

O requisito **CLI04** refere-se à escolha do estafeta para realizar a entrega. O objetivo é escolher o melhor estafeta para essa entrega. Inicialmente, a ideia era realizar um leilão para essa escolha, em que cada estafeta apresentava propostas de custo de serviço até o cliente aceitar. Esta ideia poderia apresentar preços mais baixos para o cliente porém, sendo este um serviço de entrega de refeições, o tempo gasto no leilão poderia ser uma desvantagem. Por isso, a ideia do leilão foi

abandonada e, de forma a manter os preços baixos, foi criada uma fórmula simples de cálculo de custo. O custo da encomenda é calculada pela distância que o estafeta terá de percorrer para realizar a entrega. Já a escolha do estafeta tem em conta dois fatores: o estafeta deve ser confiável (boa classificação) e deve ser o estafeta que possa realizar a entrega de forma mais rápida. A fórmula utilizada para chegar ao estafeta ideal é analisada e explicada no capítulo relativo à implementação.

4.4 Arquitetura

A arquitetura do protótipo a desenvolver nesta dissertação pode-se dividir em três partes distintas: a aplicação, o *backend* e o *back office*, não sendo esta última desenvolvida durante o período de dissertação, embora os modelos de dados criados tenham sido pensados para a sua existência no futuro.

A Figura 4.4 apresenta um diagrama de alto nível de todos os componentes utilizados na dissertação.

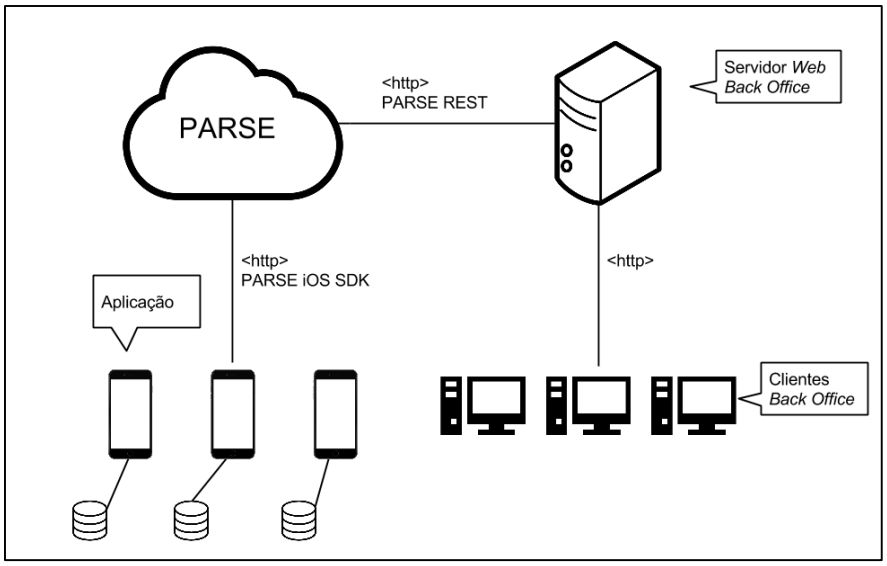

Figura 4.4: Diagrama de componentes de alto nível

O centro do sistema é o *Parse* onde é implementado o *backend*. Nele estão guardados todos os dados e implementada a lógica de negócio mais sensível.

A aplicação móvel liga-se através do SDK desenvolvido pelo *Parse* para *iOS*, que assenta sobre uma ligação HTTP. A aplicação tem a sua própria base de dados no dispositivo onde alguns dados podem ser guardados, como por exemplo, dados para início de sessão automático.

O *back office*, não foi especificado nesta dissertação, mas será uma aplicação *web* que, dependendo da linguagem em que for desenvolvido, poderá ligar-se ao Parse por um SDK que este possua ou utilizando a REST API que o *Parse* também possui. Por fim, a ligação aos clientes

do *back office* será uma ligação HTTP simples utilizando um navegador de internet instalado no dispositivo.

4.4.1 *Backend*

É no *backend* que o modelo de dados é implementado, sendo o mesmo definido sob a forma de classes, que depois pode ser reproduzido nos clientes do *backend*, que no caso deste projeto, é a aplicação *iOS*.

Sendo o *backend* desenvolvido em *Parse*, é neste componente que é implementado o modelo de dados e alguma da lógica de negócio que lida com dados mais sensíveis. O modelo de dados

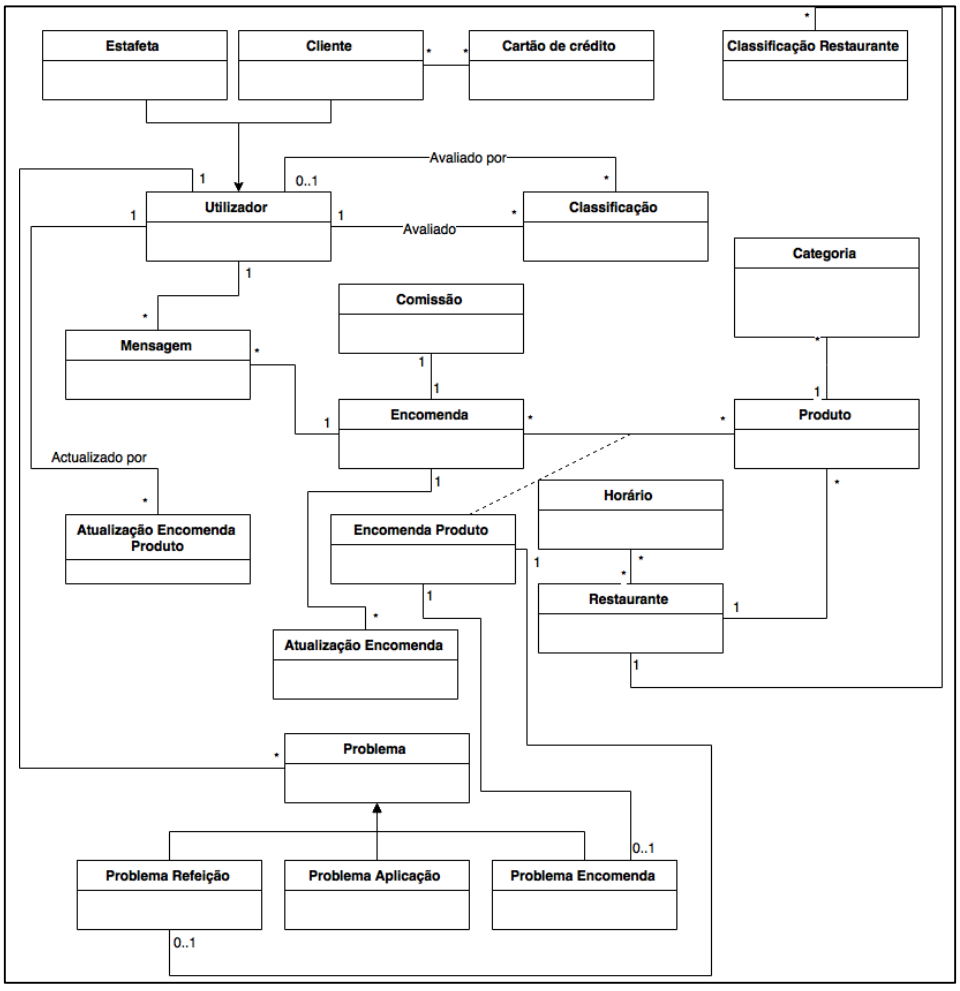

Figura 4.5: Diagrama de Classes

desenvolvido segue o diagrama de classes presente na Figura 4.5.

A classe *utilizador* é uma classe abstrata que contém os dados comuns entre *clientes* e *estafetas*. A classe *cliente* guarda informação sobre o seu *cartão de crédito*, nomeadamente a validade e os últimos quatro dígitos do cartão gerado pelo serviço de pagamentos, utilizado para a realização dos mesmos.

Uma das classes fundamentais do sistema é a classe *encomenda*, contém todos os dados da encomenda, desde a sua data de início e fim, o seu estado atual, o cliente e o estafeta, a morada de entrega, o código de confirmação, o restaurante, a localização da entrega (coordenadas GPS), as distâncias entre estafeta – restaurante e restaurante – cliente, dados relacionados com pagamentos (comissão a pagar ao estafeta e informação se o cliente já realizou o pagamento), entre outros dados importantes para o sistema. A *encomenda* está relacionada com outra classe, a *atualização de encomenda*. Esta classe guarda todas as alterações que existem ao nível do estado da encomenda, a localização do estafeta no momento da atualização e também a hora a que essa atualização é realizada.

A classe *restaurante* representa os restaurantes no sistema, onde contém informações importantes como a sua localização e o seu número de telefone. Esta classe relaciona-se com os *produtos*. Um *produto* pertence a um *restaurante* e a uma *categoria* e nas suas informações inclui um preço e um nome. O *produto* é relacionado com a encomenda através da classe *encomenda produto*. Nesta classe existe a quantidade que o *cliente* deseja desse *produto*, observações que deseja que sejam lidas pelo *estafeta* e o preço real, este preço real é o preço que o estafeta encontra no *restaurante* que pode ser diferente do preço presente na aplicação. Se esse preço for diferente, o estafeta fará uma alteração que levará à criação de um objeto da classe *atualização encomenda produto*. Esta classe tem como objetivo registar todas as alterações que possam existir nos *produtos da encomenda*.

Um ponto fundamental é a *classificação***,** tem um *utilizador* avaliado e pode ter ou não um *utilizador* avaliador (se por algum motivo a empresa quiser penalizar um *estafeta* pode adicionar uma avaliação negativa ao mesmo) e tem também uma *encomenda* relacionada. O *valor* da *classificação* varia entre um e cinco. Da mesma forma que os *utilizadores* se podem avaliar também podem avaliar um *restaurante*. A *classificação* tem um avaliador e está relacionada também com uma *encomenda*. Esta classificação apresenta também um valor que varia entre um e cinco.

De forma a existir uma via de comunicação entre o *estafeta* e o *cliente***,** existe a classe *mensagem* que guarda o conteúdo de uma mensagem, um recetor e um emissor.

Os *utilizadores* podem também relatar problemas com a encomenda, refeição ou com a aplicação. Dessa forma existem as classes respetivas que permitem guardar toda essa informação.

4.4.2 Aplicação

A arquitetura da aplicação pode ser dividida em três partes, a aplicação para o cliente, a aplicação para o estafeta e a parte comum. Embora seja apenas uma aplicação, esta comporta-se de forma diferente dependendo do papel do seu *utilizador* (*estafeta* ou *cliente*) e essa divisão foi devidamente implementada na arquitetura da aplicação.

A Figura 4.6 apresenta, de uma forma simplificada, a arquitetura da aplicação. A componente *comum* contém todas as funcionalidades comuns entre as duas outras componentes da aplicação. Contém as *constantes* onde estão implementados *tipos de dados* sobre a forma de

enumerações que representam estados de vários objetos no sistema e outras constantes importantes para o bom funcionamento da aplicação.

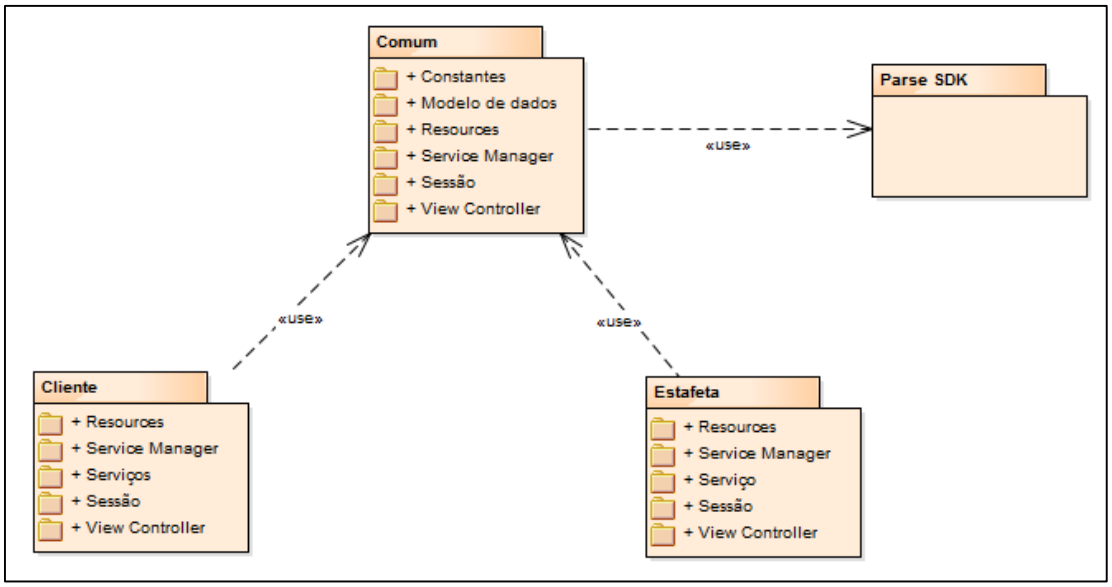

Figura 4.6: Diagrama de Componentes da Aplicação

O modelo de dados é definido também na componente *comum,* sendo implementado na mesma estrutura de classes existente no *backend* e, recorrendo ao SDK do *Parse*, é realizada a sincronização entre o modelo de dados da aplicação e o do *Parse.*

Os *resources* comuns, como algumas imagens, por exemplo, estão também presentes nesta componente.

O componente *Service Manager* implementa o padrão *Singleton*. Este padrão é utilizado para situações em que só deve existir uma instância de uma classe [Lore10], e no caso do *Service Manager*, isso é útil pois esta classe é responsável por fazer a ligação entre a aplicação e o SDK do *Parse*, de forma a abstrair ao máximo a utilização do *Parse* na aplicação. Esta componente é responsável por fornecer métodos para manipular objetos do *Parse* (criar, pesquisar e apagar) e por implementar a lógica de negócio presente na aplicação.

O componente *sessão* implementa também o padrão *Singleton* de forma a guardar os dados da sessão atual na aplicação. Por fim, o componente *view controller* implementa as vistas comuns da aplicação (por exemplo *ecrã inicial* e *registo*).

Os outros dois grandes componentes implementados, *Cliente* e *Estafeta* utilizam os componentes equivalentes presentes na parte *comum* adaptando as suas necessidades. Apresentam um componente que não se encontra no componente comum, o *serviço*. O componente *serviço* contém implementações de serviços que estão a ser executados em segundo plano na aplicação. Esses serviços são utilizados para monitorizar, por exemplo, o estado de uma encomenda no caso do *cliente* ou para verificar se existem pedidos de entregas no caso de um *estafeta*.

Capítulo 5

Implementação

Neste capítulo são analisados alguns pormenores da implementação. Começa-se pela metodologia utlizada neste processo e analisam-se alguns dos algoritmos e desenvolvimentos para atingir os requisitos da aplicação.

5.1 Metodologia

Na fase de implementação optou-se por dividir o trabalho por módulos. Essa divisão teve em conta as narrativas de utilização e a sua prioridade. Cada módulo passou depois por várias fases. Inicialmente foi realizada a pesquisa de soluções já existentes que poderiam resolver os

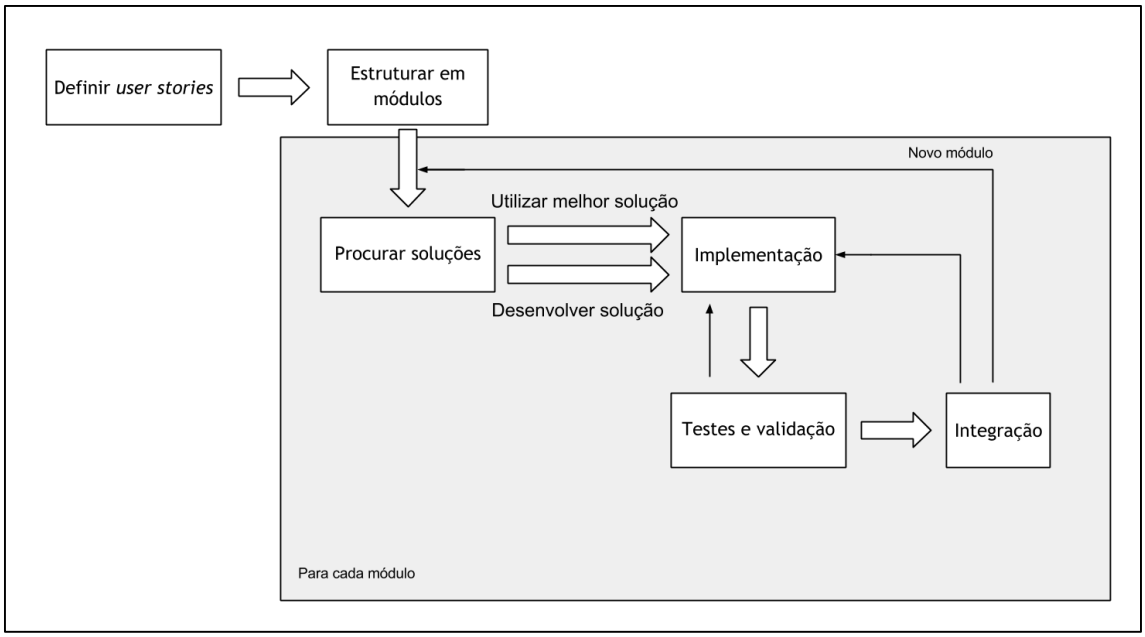

Figura 5.1: Metodologia

problemas do módulo. Se existirem soluções escolhe-se a melhor e adapta-se ao problema. Se não se encontrou nenhuma solução, foi necessário desenvolver uma para o problema. Após a implementação, o resultado foi testado e avaliado. Após a solução cumprir os requisitos integrou-se com os restantes módulos e passou-se para o próximo módulo. A Figura 5.1 apresenta um esquema deste processo.

As secções seguintes demostram alguns dos módulos criados neste processo, focando os problemas que se abordaram e a resolução encontrada para os mesmos.

5.2 Privacidade de dados

Para um utilizador poder usufruir de uma aplicação deste tipo tem de fornecer dados pessoais que na maioria dos casos não os quer ver exposto. Esses dados têm de ser devidamente protegidos de forma a não ficarem vulneráveis.

O *Parse* possui mecanismos que permitem definir autorização de acesso a dados podendo ser aplicados de duas formas. A primeira é recorrendo ao *Class-Level Permissions* (CLP) que permite definir uma lista de permissões (de criação, de escrita e de leitura) ao nível da classe para utilizadores ou grupos de utilizadores com determinados cargos – *Roles* [Pars15b]. A outra opção é utilizar *Object-Level Access Control* recorrendo a *Access Control Lists* (ACL) que define listas de permissões semelhantes às utilizadas no CLP mas ao nível do objeto. [Pars15b]

Mesmo com estes mecanismos que o *Parse* já tem implementados não é possível garantir a privacidade dos dados seguindo o modelo de dados presente na Figura 4.5. Isso acontece porque existem casos em que dentro do mesmo objeto existem dados que não devem estar visíveis a determinados utilizadores, mas são necessários a outros. Como o *Parse* possui apenas controlo de acessos ao nível do objeto não é possível determinar uma lista de acessos por atributos, tendo de se expor todos os dados.

De forma a manter a privacidade dos dados, a solução encontrada passou por criar novas classes com os dados sensíveis, e nessas classes os objetos criados terem permissões diferentes. Na prática o que acontece é que a classe principal passa a possuir atributos que são apontadores para outras classes que possuem um conjunto de permissões diferentes. Deste modo, quando um utilizador consultar esse objeto o *Parse* irá devolver todos os atributos da classe mais os apontadores para essas novas classes, isto se o utilizador tiver permissões de leitura dos mesmos. Se não possuir permissões de leitura sobre esses objetos das novas classes não existirá referência à existência desse atributo[2].

[2] Num só pedido é possível obter todos os dados dos apontadores existentes num objeto, basta para isso indicá-lo no mesmo pedido. Mas a verificação de permissões é respeitada da mesma forma.

5.2.1 Alguns Exemplos

A classe *cliente* é um bom exemplo porque apresenta um grande conjunto de dados que não devem ser exposto a todos os utilizadores. Desta forma foram criadas duas novas classes para além da classe original. Foi criada a classe com os dados privados do cliente, ao qual só ele tem acesso e uma classe com dados que quer partilhar com um estafeta, mas apenas quando este lhe está a prestar o serviço.

A Figura 5.2 apresenta sobre a forma de diagrama simplificado essas alterações. Assim as três classes ficam acessíveis aos *gestores de sistema* por via CLP com controlo total. Na classe

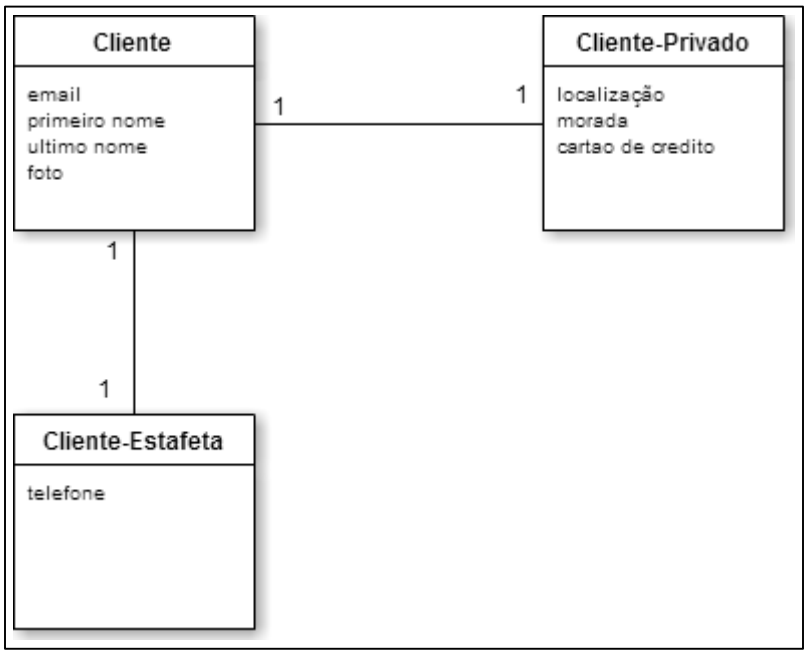

Figura 5.2: Alterações classe Cliente

Cliente é configurada uma ACL onde é dado controlo total sobre o objeto ao próprio cliente e permissões de leitura aos estafetas que estejam ou já tenham realizado alguma entrega a esse cliente[3]. Já a classe *Cliente-Privado* só o cliente é que lhe pode aceder tendo todas as permissões sobre o objeto. Na classe *Cliente-Estafeta*, à semelhança das restantes, é o próprio cliente que tem controlo total sobre o objeto, tendo apenas permissões de leitura o estafeta que esteja nesse momento a realizar uma entrega ao cliente.

A classe *encomenda*, uma das classes fundamentais do sistema, utiliza um sistema semelhante ao da classe *cliente* mas com algumas particularidades. De forma semelhante à classe *cliente* dá permissões totais aos *gestores de sistema* via CLP. Já a classe base só fica acessível a dois utilizadores, ao cliente e ao estafeta (configurado via ACL). Contém também classes privadas

[3] De forma a simplificar o texto em alguns casos é indicado que só determinados utilizadores têm acesso a uma classe porém, os *gestores de sistema* têm sempre acesso total a todas as classes.

usando um esquema semelhante à classe *Cliente*. Neste caso a classe privada contém o PIN de confirmação de encomenda que só pode ser consultado pelo cliente.

A classe *cliente* e a classe *encomenda* recorrem ao CLP e ao ACL para definir as permissões mas a classe *restaurante*, por exemplo, não necessita de recorrer ao ACL. Esta classe deve ser visível por todos, porém só pode ser alterada ou criada por *gestores de sistema*. Esta permissão é configurada por via de CLP com permissões de leitura a todos os utilizadores e permissões de escrita e criação aos utilizadores que estejam presentes no grupo de utilizadores (*Role*) *Gestores de Sistema*.

5.3 Algoritmos e processos

Todo o serviço à volta da aplicação conta com vários algoritmos que permitem o bom funcionamento da aplicação. Em alguns casos esses algoritmos otimizam a informação a apresentar ao utilizador. Já noutros implementam os processos da lógica de negócio da aplicação. As próximas secções apresentam os algoritmos e processos mais relevantes implementados.

5.3.1 Escolha de restaurantes

Um cliente da aplicação precisa de consultar uma lista de restaurantes para fazer as suas encomendas. Essa lista deve apresentar os restaurantes que realmente sejam úteis para o utilizador. Assim sendo a lista de restaurantes apresentada ao cliente é filtrada e ordenada de forma a dar uma melhor experiência de utilização.

Os restaurantes apresentados ao cliente são em primeiro lugar restaurantes perto da sua localização (a uma distancia pré-definida) ou perto de uma localização que o cliente queira pesquisar. Após este filtro é realizada uma avaliação a cada um desses restaurantes de forma a perceber se o restaurante está disponível para receber encomendas. Para estar disponível tem de estar aberto e possuir estafetas ativos (esperando encomendas) perto. Por fim a lista é ordenada em dois níveis. Primeiro aparecem os restaurantes disponíveis (abertos e com estafetas perto), depois os restaurantes abertos mas sem estafetas por perto e por último os restaurantes fechados. Estes últimos dois grupos são listados mas não é possível realizar encomendas aos mesmos. Dentro de cada um dos três grupos os restaurantes são ordenados por classificação, os restaurantes mais cotados aparecem em primeiro lugar na lista.

5.3.2 Previsão do tempo de espera

A previsão do tempo de espera é apresentado ao cliente de forma a este ter uma melhor noção da hora a que será realizada a entrega. Esta previsão é calculada recorrendo ao seguinte conjunto de variáveis:

- **Velocidade média do estafeta (VE):** a velocidade média do estafeta é calculada através de uma média aritmética da velocidade do estafeta nos percursos estafeta – restaurante e restaurante – local de entrega de todas as estregas já realizadas pelo estafeta;

- **Distância do estafeta ao restaurante (DER):** esta distância é calculada recorrendo à API do *Google Maps* [Goog15], utilizando o percurso mais curto quando percorrido de automóvel.

- **Distância do restaurante ao local de entrega (DRE):** esta distância é calculada de forma semelhante à distância do estafeta ao restaurante.

- **Tempo médio de espera no restaurante por produto (TEP):** este tempo é calculado através da média aritmética do tempo que o estafeta esperou no restaurante pelo número de produtos de todas as encomendas já realizadas ao restaurante em questão. Existem produtos que não entram para este cálculo, por não terem um tempo de preparação associado (por exemplo as bebidas).

- **Tempo de confeção (TC):** este tempo é introduzido pelo próprio estafeta na aplicação, o estafeta deve indicar este valor se ao entrar em contacto com o restaurante lhe for indicado o tempo de confeção.

- **Número de produtos (NP):** número de produtos que afetam o tempo de espera no restaurante.

A previsão do tempo de espera é apresentada ao cliente em diferentes situações, em que, dependendo da situação, a equação de cálculo é diferente.

A primeira vez que o tempo de espera é apresentado ao cliente é no momento em que este está a confirmar a sua encomenda. Neste momento ele ainda não tem estafeta atribuído, dessa forma são analisados os estafetas que são elegíveis para realizar essa entrega e é apresentado o tempo do estafeta que demoraria menos tempos a entregar a encomenda e o que demoraria mais tempo. O cálculo para cada um dos estafetas recorre à Equação 5.1.

$$tempoEspera = (VE \times DER) + (NP \times TEP) + (VE \times DRE) \quad (5.1)$$

Após ser encontrado um estafeta para realizar a entrega o tempo previsto de espera é apresentado no ecrã, sobre a forma da hora prevista de entrega. O cálculo dependerá do estado da encomenda. Enquanto o estafeta se está a dirigir para o restaurante a equação de cálculo pode apresentar uma de duas formas. Se o estafeta tiver contactado o restaurante e souber o tempo de confeção das refeições a equação utilizada será a equação 5.2, porque enquanto o estafeta realiza a viagem em direção ao restaurante a refeição já está a ser preparada. Se não tiver introduzido o tempo de confeção a equação utilizada será a equação 5.1.

$$tempoEspera = \max((VE \times DER), TC) + (VE \times DRE) \quad (5.2)$$

O resultado das duas fórmulas é o tempo previsto de espera a partir do momento que o estafeta indica que está a dirigir-se para o restaurante.

Chegando ao restaurante, a componente das equações referentes à viagem do estafeta até ao restaurante é retirada. Porém, na equação 5.1, o resultado indica o tempo de espera a partir do momento que o estafeta chega ao restaurante. Já na equação 5.2 continua a ser a partir do momento que o estafeta indica que se está a dirigir para o restaurante.

Quando o estafeta sai do restaurante o cálculo é realizada recorrendo a uma só equação, já sem as componentes referentes à viagem do estafeta até ao restaurante e o tempo de espera no restaurante. A equação 5.3 reflete o cálculo neste estado da encomenda.

$$tempoEspera = VE \times DRE \tag{5.3}$$

A partir do momento que o estafeta chega ao local de entrega, o tempo previsto de espera deixa de ser apresentado ao cliente.

5.3.3 Processo de escolha de estafetas

Quando um cliente cria uma encomenda é aberto o processo de escolha de estafetas. Este processo tem três fases distintas.

Na primeira fase deste processo são escolhidos os *estafetas elegíveis* para realizar a entrega. Para um estafeta ser considerado elegível para realizar uma entrega não pode estar a realizar uma outra entrega mas tem de estar em *modo de espera*. Desse modo os estafetas apresentam uma variável de estado que pode conter um de três valores:

- **Desligado:** Não se encontra disponível para realizar entregas.
- **Em encomenda:** Encontra-se neste momento a realizar uma entrega.
- **Em espera:** Encontra-se à espera de pedidos de encomendas.

Um estafeta para ser considerado elegível deve então estar no modo *em espera*. O outro critério implica que o estafeta se encontre perto do restaurante, tendo de estar a menos de uma distância pré-definida pelos gestores do sistema.

Após ser criada uma lista com todos os *estafetas elegíveis* inicia-se a segunda fase do processo onde é necessário ordenar os *estafetas elegíveis* de forma a contactar primeiro os que sejam uma mais-valia para o cliente. Para os ordenar é calculado um valor através de uma função heurística para cada um dos *estafetas elegíveis*, com base no seguinte conjunto de variáveis:

- **Distância do estafeta ao restaurante (DER):** esta distância é calculada recorrendo à API do *Google Maps* [Goog15], utilizando o percurso mais curto quando percorrido de automóvel;
- **Velocidade média do estafeta (VE):** a velocidade média do estafeta é calculada através de uma média aritmética da velocidade do estafeta nos percursos estafeta –

restaurante e restaurante – local de entrega, de todas as estregas já realizadas pelo estafeta;

- **Classificação do estafeta (CE):** a média aritmética das classificações atribuídas pelos clientes ao estafeta em questão.

A equação 5.4 representa a função heurística que devolve o valor pelo qual os estafetas devem ser ordenados. Nessa equação, para os estafetas com classificação iguais ou superiores a 4, a função calcula o tempo da deslocação até ao restaurante. Já para os estafetas com classificação inferior a 4, o tempo de deslocação é multiplicado pelo resultado da diferença de cinco com a sua classificação. O melhor estafeta é o estafeta com o valor menor.

$$f = \begin{cases} DER \times VE, & CE >= 4 \wedge CE <= 5 \\ (5 - CE) \times (DER \times VE), & CE < 4 \wedge CE >= 0 \end{cases} \quad (5.4)$$

O objetivo desta função heurística é não diferenciar estafetas com classificação superiores a 4, sendo já considerados estafetas de qualidade, escolhendo o estafeta que realizará o percurso mais rapidamente. Por outro lado a heurística também tem o objetivo de penalizar os estafetas com classificação inferior a 4, aumentado o resultado da função heurística. Para isso é multiplicado o tempo da deslocação até ao restaurante com o resultado da diferença da sua classificação com a classificação máxima, 5.

Tendo os *estafetas elegíveis* devidamente ordenados pelo valor calculado anteriormente inicia-se a última fase deste processo. Esta última fase é onde realmente ocorre a atribuição de estafetas. Este processo implica contactar o melhor estafeta, aguardar a sua resposta e se a resposta for afirmativa é atribuído esse estafeta à encomenda. Caso seja negativa repete-se o processo para o segundo melhor estafeta e assim sucessivamente.

Para implementar esse processo foi criada a entidade *estafeta elegível*, que além de conter o resultado da função heurística já referida, contém também uma variável de estado que pode assumir os seguintes valores:

1. **Aceite**: O estafeta aceitou realizar a encomenda;
2. **Recusado**: O estafeta recusou realizar a encomenda;
3. **Esperando resposta**: O estafeta foi contactado e ainda não respondeu, mas está ainda dentro do tempo previsto de resposta;
4. **Não contactado**: O estafeta não foi contactado;
5. **Último contacto**: O estafeta foi contactado e caso não responda será excluído da lista;
6. **Sem resposta**: O estafeta não respondeu.

Os estafetas são ordenados numa lista, primeiro pelo estado (*Aceite* em primeiro) e dentro de cada estado são ordenados pelo valor da função heurística do estafeta. Assim sendo, inicialmente todos os *estafetas elegíveis* apresentam o estado 4 – *Não contactado*. É verificado o estado do primeiro elemento iniciando-se a máquina de estados representada na Figura 5.3.

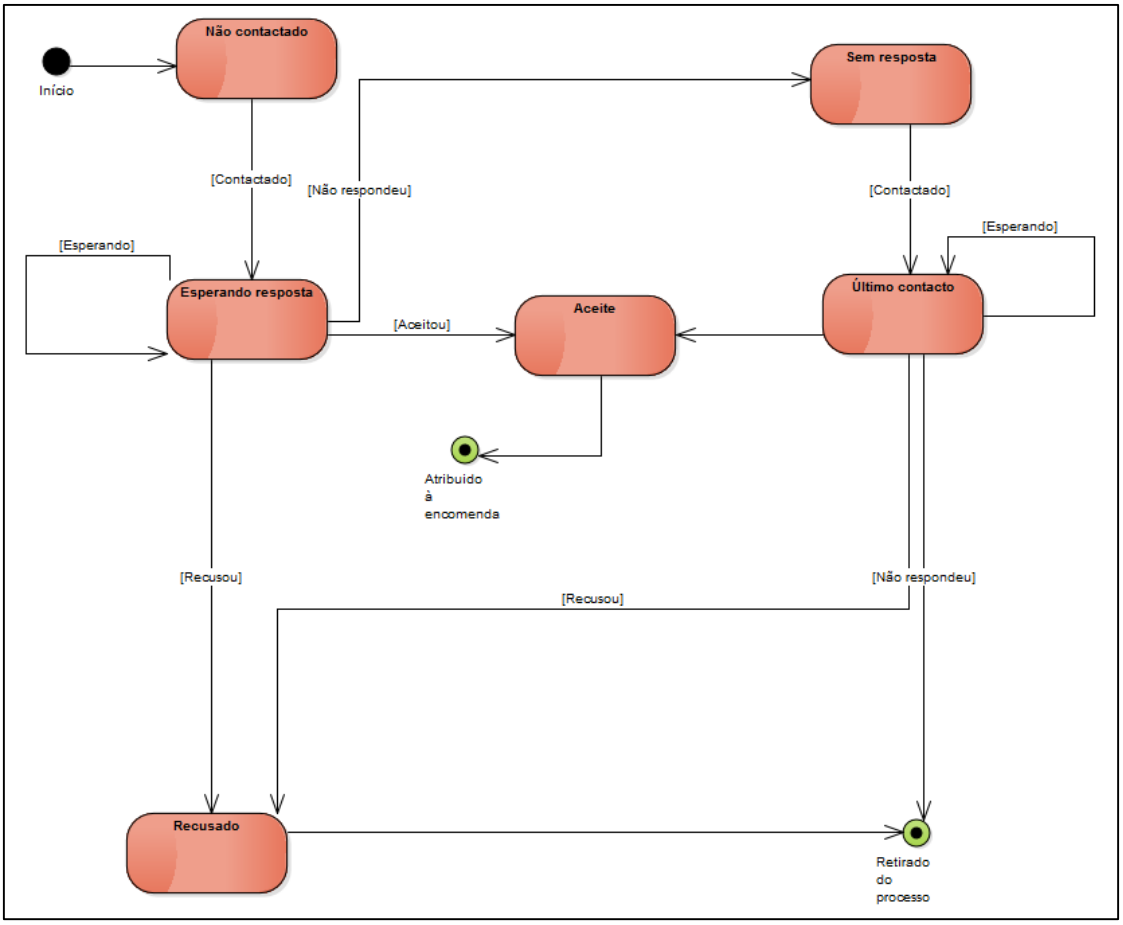

Figura 5.3: Máquina de estados *Estafeta Elegível*

O seu funcionamento é resumido da seguinte forma:

- Se o estado foi **Aceite** o estafeta é atribuído à encomenda e o processo termina;
- Se o estado for **Recusado** o estafeta é removido da lista e é analisado o próximo elemento da mesma;
- Se o estado for **Esperando resposta** podem acontecer duas situações diferentes. O estafeta tem um tempo (determinado pelo gestor do sistema) para responder ao sistema se aceita ou recusa realizar a entrega. Se esse tempo ainda não tiver sido ultrapassado o estado mantém-se e é indicado que a encomenda ainda não tem estafeta atribuído. Se já tiver terminado é alterado o estado para **Sem resposta**. A lista é novamente ordenada e volta-se a repetir o processo para o primeiro elemento;

- Se o estado for **Não contactado** isso indica que esse estafeta ainda não foi contactado. É então contactado sendo guardada a hora de contacto e passando o estado para **Esperando resposta**;

- Se o estado for **Último contacto** o processo é semelhante ao estado **Esperando resposta** com a diferença de que se o tempo já tiver terminado esse estafeta é eliminado da lista;

- Se o estado for **Sem resposta** o processo é semelhante ao estado **Não contactado** com a diferença que o estado é alterado para **Último contacto**;

- Se a lista estiver vazia isso significa que não foi possível encontrar um estafeta para realizar a encomenda.

Ao nível mais técnico este processo é implementado recorrendo ao *cloud code* do *Parse*, nomeadamente as *cloud functions* e aos *backgrounds jobs*. A primeira e segunda parte do processo estão contidas numa *cloud function* que é executada quando a encomenda é criada pelo *cliente.* Esta função irá criar objetos *estafeta elegível* associados à encomenda.

A ser executado em segundo plano, no *Parse*, existe um processo que funciona como um *cron job*. Um *cron job* executa uma determina tarefa periodicamente [Heng14]. Neste caso executa uma função que verifica se existem encomendas em processo de escolha de estafetas e para cada um delas executa uma *cloud function* onde se encontra implementada a terceira fase do processo.

As aplicações vão verificando também periodicamente se existem alterações, utilizando classes *Singleton* que executam pedidos em segundo plano. O cliente verifica se o estado da encomenda foi alterado e o estafeta verifica se existe algum objeto e*stafeta elegível* que o referencie e esteja no estado *Esperando resposta* ou *Último contacto***.**

5.3.4 Encomenda

A encomenda é a entidade que contém toda a informação relativa à mesma, e de forma semelhante a outras entidades, com processos ou algoritmos associados, é também controlada com uma variável de estado. Nesta entidade a variável de estado é utilizada para acompanhar o ponto de situação da encomenda e também para controlo do fluxo da aplicação, podendo o utilizador sair da aplicação e voltar a entrar sem perder o seu estado.

Assim sendo, a encomenda conta com um grande conjunto de estados possíveis:

- **À espera de estafetas**: estado inicial da encomenda. Neste estado a encomenda encontra-se em processo de escolha de estafeta;

- **Encontrado estafeta**: um estafeta aceitou realizar a encomenda;

- **A encomenda foi alterada**: o estafeta alterou a encomenda, por exemplo, o preço de um produto. A encomenda fica pendente da decisão do cliente de aceitar ou recusar a alteração;
- **O cliente aceitou a alteração**: o cliente aceitou a alteração realizada pelo estafeta e o processo da entrega da encomenda pode prosseguir;
- **O estafeta está a ir para o restaurante**: o estafeta indicou que iniciou a viagem para o restaurante;
- **O estafeta chegou ao restaurante**: o estafeta indicou que chegou ao restaurante;
- **A encomenda foi alterada (no restaurante)**: O estafeta alterou a encomenda no restaurante;
- **O cliente aceitou a alteração (no restaurante)**: O cliente aceitou a alteração que o estafeta realizou no restaurante e o processo da entrega da encomenda pode prosseguir;
- **A encomenda foi paga no restaurante**: O estafeta pagou a encomenda no restaurante e enviou o comprovativo;
- **O estafeta encontra-se a dirigir-se para o local de entrega**: O estafeta indicou que saiu do restaurante. Está a fazer a viagem para se encontrar com o cliente;
- **O estafeta chegou ao local da entrega**: O estafeta indicou que chegou ao local, estando à espera do cliente;
- **O estafeta enviou uma fotografia do local onde se encontra**: O estafeta não está a encontrar o cliente, e enviou uma fotografia onde é facilmente percetível a sua localização;
- **A encomenda foi entregue com sucesso**: A encomenda foi entregue e o cliente introduziu o código de confirmação;
- **A entrega falhou**: A entrega falhou e o motivo da falha ainda não foi definido;
- **A entrega falhou e esta a ser analisado o motivo da falha**: A encomenda falhou e um gestor do sistema irá analisar e concluir o motivo da falha;
- **A entrega falhou e o cliente terá de efetuar o pagamento completo**: Foi considerado que a falha foi culpa do cliente, devendo ser realizado o pagamento da encomenda;
- **A entrega falhou e o estafeta é responsável por todos os custos**: Foi considerado que a culpa da falha foi do estafeta, não receberá nenhum pagamento pelo serviço prestado

ao estafeta, podendo sofrer mais tarde uma penalização por parte da empresa. O cliente não realizará nenhum pagamento nesta situação;

- **Não foi encontrado estafeta para o pedido**: A encomenda falhou porque não foi encontrado nenhum estafeta para o serviço;
- **O cliente rejeitou uma alteração**: A encomenda falhou porque o cliente rejeitou uma alteração na encomenda, não será cobrado nenhum custo ao cliente;
- **A encomenda foi cancelada pelo cliente sem custos:** O cliente cancelou a encomenda quando ainda não tinha estafeta atribuído, desse modo não será cobrado qualquer custo;
- **A encomenda foi cancelada pelo cliente após atribuição de um estafeta**: O cliente cancelou a encomenda, quando já tinha um estafeta atribuído. É cobrado o valor total da encomenda;
- **O restaurante encontra-se fechado**: A encomenda é cancelada porque o restaurante se encontra fechado.

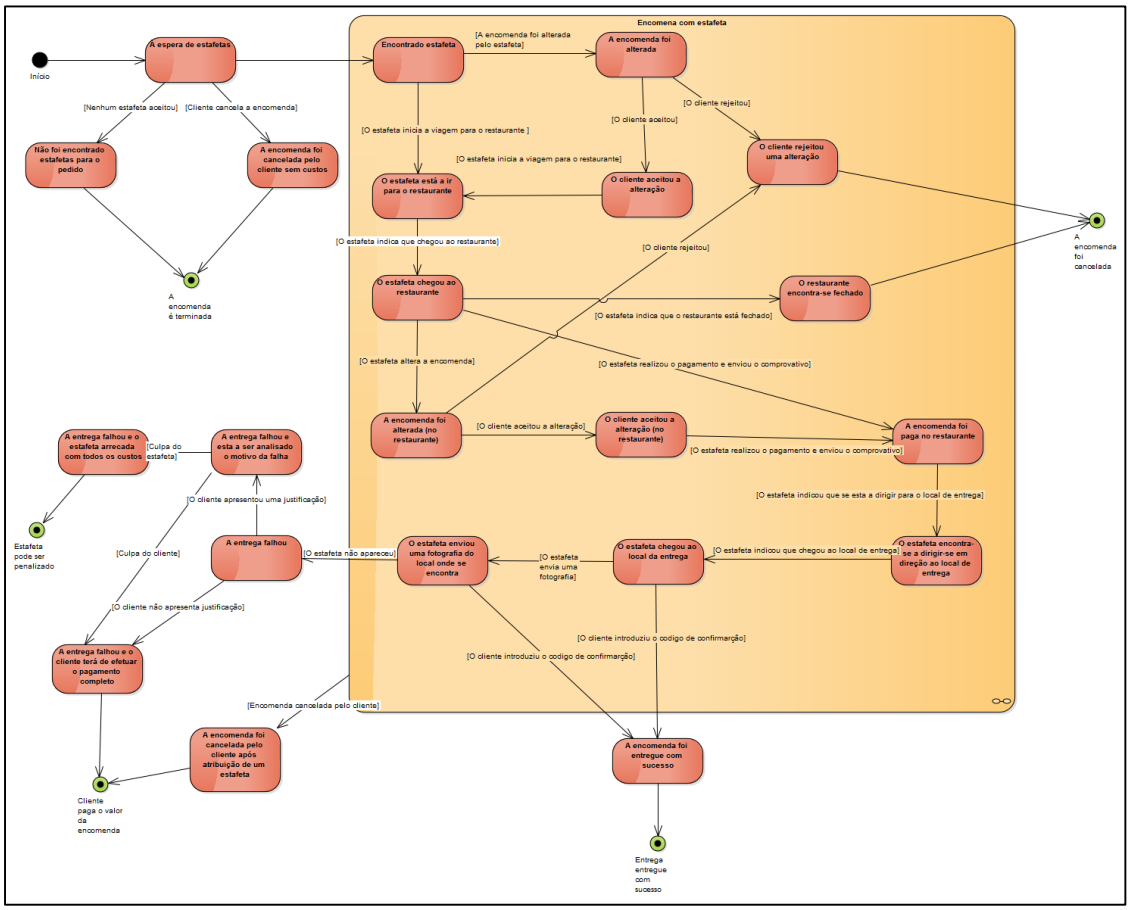

Figura 5.4: Máquina de estados *encomenda*

Na encomenda, a validade de uma alteração de estado depende do estado anterior. Quando isto acontece pode levar a problemas de concorrência. De forma a evitá-los foi implementada uma máquina de estados que é executada no *Parse* antes de guardar uma alteração numa encomenda, recorrendo aos *triggers before save* que são executados antes do objeto ser guardado. O objetivo destes *triggers* é a validação de dados, que neste caso valida a transição de estado da encomenda.

A Figura 5.4 apresenta essa mesma máquina de estados cujo funcionamento é explicado com mais detalhe de seguida:

- Estando no estado **À espera de estafetas** o estado pode ser alterado em três situações:
    - Foi encontrado um estafeta e esse estafeta é atribuído à encomenda sendo o estado alterado para **Encontrado estafeta**;
    - Não foi encontrado estafeta, sendo deste modo a encomenda cancelada e o estado alterado para **Não foi encontrado estafetas para o pedido**;
    - O cliente cancelou a encomenda, e como não tem estafeta atribuído este cancelamento não tem custo para o cliente, sendo o estado alterado para **A encomenda foi cancelada pelo cliente sem custos**.
- A partir do momento que existe um estafeta atribuído até a encomenda ser entregue ou falhar, se o cliente cancelar a encomenda terá de arcar com todos os custos da mesma. Nesta situação o estado é alterado para **A encomenda foi cancelada pelo cliente após atribuição de um estafeta**;
- Quando a encomenda se encontra no estado **Encontrado estafeta** podem acontecer duas situações:
    - O estafeta pode entrar em contacto com o restaurante e perceber que tem de alterar a encomenda. O estado é então alterado para **A encomenda foi alterada**;
    - O estafeta pode indicar que está a partir em direção ao restaurante, sendo o estado alterado para **O estafeta está a ir para o restaurante**.
- No estado **O estafeta está a ir para o restaurante** o estafeta pode alterar o estado quando chega ao restaurante, indicando isso mesmo. A encomenda passa ao estado **O estafeta chegou ao restaurante**;
- Estando no restaurante o estafeta pode alterar o estado da encomenda para três estados possíveis:
    - Se o restaurante estiver fechado o estafeta deve indicar isso mesmo e a encomenda é cancelada, sendo o estado alterado para **O restaurante encontra-se fechado**;

- O estafeta pode ter de alterar os valores da encomenda, alterando o estado para **A encomenda foi alterada (no restaurante)**;

- O estafeta pode indicar que efetuou o pagamento da refeição, enviando o comprovativo. Nesse caso o estado é alterado para **A encomenda foi paga no restaurante**.

- Quando uma encomenda é alterada, seja antes de o estafeta estar no restaurante ou depois, os estados que a encomenda pode tomar são semelhantes:

  - O cliente pode aceitar a alteração, alterando o estado para **O cliente aceitou a alteração** ou **O cliente aceitou a alteração (No restaurante)** dependendo do local de onde foi realizada a alteração;

  - O cliente pode recusar a alteração da encomenda, levando a que a encomenda seja cancelada. O estado é alterado para **O cliente rejeitou uma alteração**.

- Após o estafeta efetuar o pagamento e enviar o respetivo comprovativo, o próximo estado que a aplicação pode tomar é o estado que indica que o estafeta está a dirigir-se para o local da entrega: **O estafeta encontra-se a dirigir-se para o local de entrega**;

- Após iniciar a viagem até ao cliente o estafeta só pode alterar a encomenda quando chegar ao local da entrega, ficando a encomenda com o estado **O estafeta chegou ao local da entrega**;

- Estando a encomenda no estado **O estafeta chegou ao local da entrega** esta pode transitar para os seguintes estados:

  - **A encomenda foi entregue com sucesso** se o cliente introduzir o código no *smartphone* do estafeta confirmando assim a receção;

  - **O estafeta enviou uma fotografia do local onde se encontra**: se o estafeta não estiver a encontrar o cliente pode enviar uma fotografia com o local onde se encontra. A partir desse momento o cliente tem cinco minutos para se dirigir ao local da entrega.

- Após o estafeta enviar a fotografia a encomenda pode assumir dois estados:

  - O cliente não apareceu durante os cinco minutos de espera, e o estafeta cancelou a encomenda passando para o estado **A entrega falhou**;

  - O cliente apareceu e confirmou a encomenda, introduzindo o código no *smartphone* do estafeta sendo a encomenda alterada para **A encomenda foi entregue com sucesso**.

- Quando o estafeta cancela a encomenda por o cliente não aparecer a encomenda pode assumir um de dois estados:
    - O cliente tem direito de se defender apresentado uma justificação, passando o estado para **A entrega falhou e está a ser analisado o motivo da falha**;
    - O cliente pode aceitar que a encomenda não foi entregue por sua causa e nesse caso o estado é alterado para **A entrega falhou e o cliente terá de efetuar o pagamento completo** sendo efetuado o pagamento da encomenda;
- Quando a falha na entrega da encomenda vai para análise esta pode evoluir para um de dois estados:
    - O gestor de sistema considera que o cliente deve fazer o pagamento da encomenda alterando o estado para **A entrega falhou e o cliente terá de efetuar o pagamento completo**;
    - O gestor de sistema considera que o estafeta falhou e o cliente não deve pagar nenhum custo da encomenda. Alterando o estado para **A entrega falhou e o estafeta é responsável por todos os custos**.

5.3.5 Resolução de conflitos

Na secção 4.3 foram apresentadas algumas ideias para resolver os conflitos entre utilizadores. Essas ideias não apresentavam grandes desafios de implementação, pois o objetivo é recolher o maior número de dados que facilite a resolução de conflitos por parte de um gestor de sistema, e não a resolução automática dos mesmos problemas. Uma das ideias para a resolução de conflitos, passava por permitir que o estafeta enviasse uma fotografia com a localização para o cliente, e depois aguardasse no local durante cinco minutos. Durante esse período de tempo a localização seria monitorizada. De forma a minimizar a produção de dados, a localização capturada só é enviada para o servidor quando a alteração da localização é significativa. O estafeta é também notificado que não se encontra no local da entrega. Esta informação é guardada e pode ser visualizada por um gestor de sistema em caso de conflito, de forma a resolvê-lo. Outras medidas foram tomadas, essencialmente ao nível da alteração de dados.

Uma dessas medidas é capturar a localização do estafeta sempre que este efetua uma alteração do estado da encomenda. Para que o gestor de sistema na eventualidade de ter de resolver um conflito poder compreender melhor o percurso do estafeta. Neste ponto foi ponderado fazer uma captura em tempo real da localização durante todo o percurso do estafeta porém isso iria levar a um acréscimo no consumo de bateria, e para alguns estafetas isso seria um problema, pois além da necessidade de utilizar o GPS também teria de utilizar uma ligação à Internet com

custos para que os dados recolhidos fossem enviados para o servidor de modo a serem armazenados.

O gestor de sistema pode também recorrer às mensagens trocadas entre os utilizadores em conflito de forma a perceber se existe informação útil para resolver o conflito. Na implementação foram dadas permissões aos gestores do sistema para conseguir visualizar essa informação.

As alterações realizadas à encomenda são também guardadas, criando um histórico de alterações que permite serem consultadas pelo gestor do sistema.

5.4 Pagamentos

Como já foi referido, o serviço de pagamentos escolhido para ser utilizado na aplicação foi o *Switch*, que é produto de uma *spin-off* da empresa de acolhimento.

Para processar um pagamento com o *Switch*, no contexto da aplicação desenvolvida nesta dissertação, o primeiro passo é a criação de um cartão virtual. Neste processo o dono do cartão

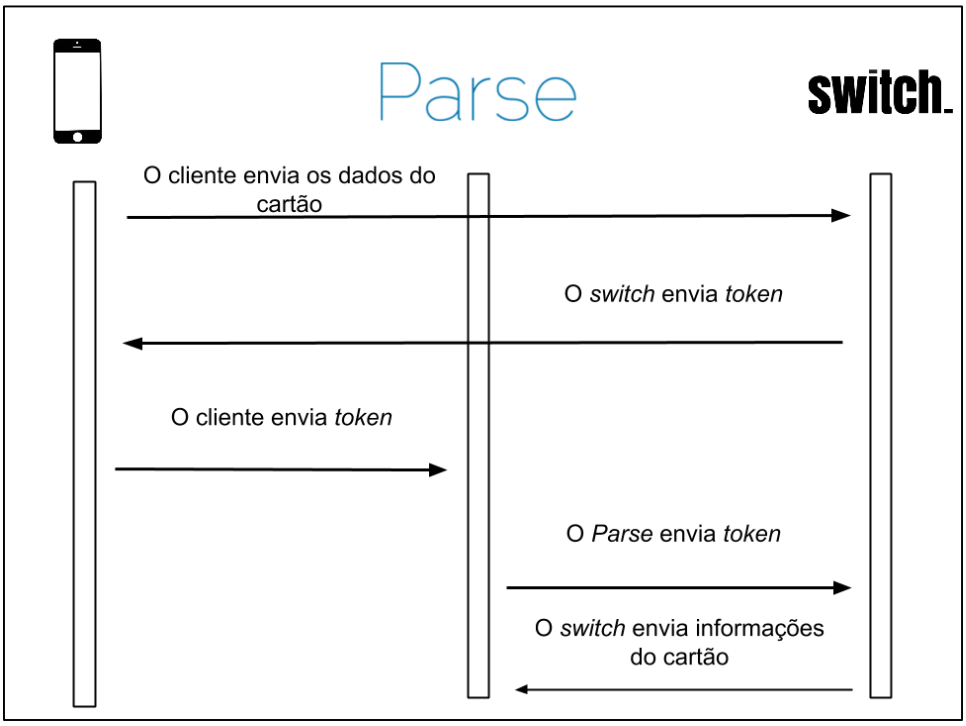

Figura 5.5: Criação de um cartão virtual

deverá introduzir todos os dados do cartão (nome do dono, número do cartão, CVV e validade) na aplicação de forma a serem enviados para o *Switch*. Este processo é realizado através da API REST do *Switch*, utilizando uma *Basic access authentication* com a chave pública que o *Switch* fornece ao comerciante. Esta comunicação, por sua vez é realizada com recurso a HTTPS. Na resposta é enviado um *token* do tipo *card*.

O *token* recebido deve ser enviado para o *backend* do comerciante, neste caso para o *Parse*. Ao receber esse *token* o *backend* pode ligar-se ao *Switch*, com um processo semelhante à ligação

que é realizada na criação do *token* na aplicação, mas utilizando agora para autenticação a chave privada e o *ID* fornecidos pelo *Switch*. Nesta comunicação o *backend* deve enviar o *token* que recebeu do cliente e receberá do *Switch* os dados relativos ao cartão, nomeadamente o nome do dono do cartão, a validade, os últimos quatro dígitos do mesmo e um outro *token* que será utilizado para realizar os pagamentos. A Figura 5.5 apresenta um diagrama de sequência do processo de criação do cartão virtual.

Depois de ter o cartão criado o *backend* pode realizar os pagamentos. No contexto da dissertação os pagamentos são executados pelo *Parse* quando a encomenda atinge determinados estados. Para realizar um pagamento o servidor deve enviar para o *Switch* o *token* do cartão virtual, e o montante do pagamento. Neste ponto o *Switch* tem duas opções, pode-se apenas avaliar se é possível realizar o pagamento ou realizar de imediato o pagamento. Nesta aplicação optou-se pela segunda opção. Assim sendo, ao fazer o pedido ao *Switch* a resposta indicará se o pagamento foi bem ou mal sucedido. A Figura 5.6 apresenta um diagrama de sequência deste processo.

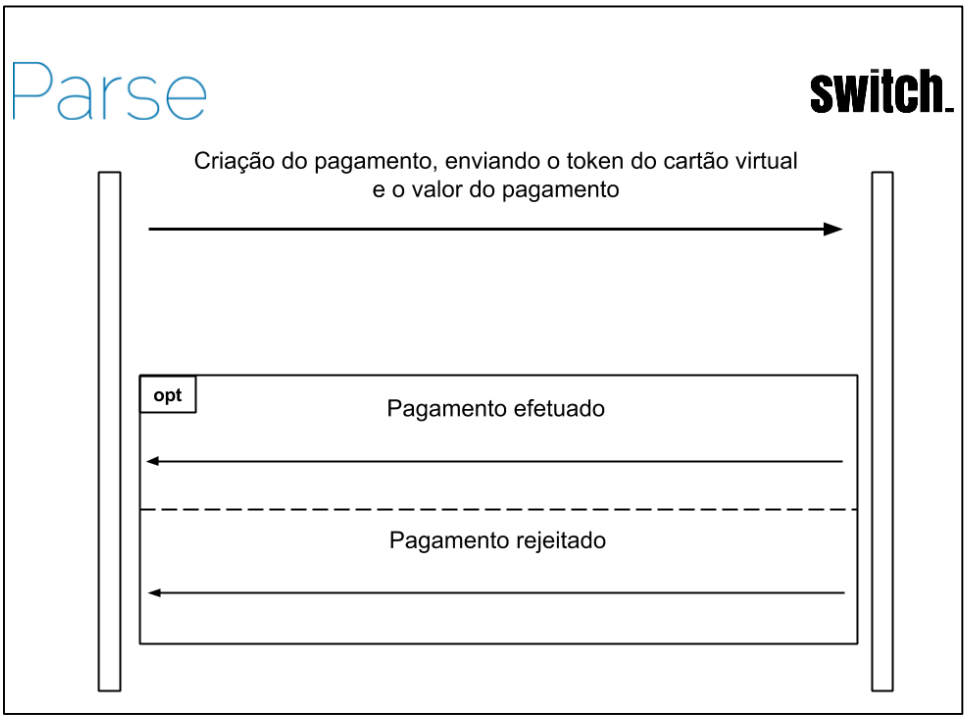

Figura 5.6: Pagamento

5.4.1 SDK e *Cloud Code Module*

A integração do sistema de pagamentos com a aplicação e com o Parse só era possível recorrendo à API REST fornecida por serviço. De forma a simplificar essa integração a futuros utilizadores foi desenvolvido um SDK de *iOS* simples para a componente cliente do serviço de pagamento e um *Cloud Code Module* do *Parse* para *backend*.

O SDK desenvolvido para *iOS* é muito simples, contém apenas dois métodos:

- *Setup*: que recebe como parâmetros a chave pública e o ambiente em que vai ser utilizado (testes ou produção).

- *CreateCardToken:* que recebe como parâmetro os dados do cartão devolvendo um objeto que contém alguma da informação do cartão juntamente com o *token* que o *Switch* criou, com os dados do cartão enviado.

A documentação deste SDK encontra-se no anexo B.
Já o módulo criado para o *Parse* apresenta os seguintes métodos:

- *Initialize*: que recebe como parâmetro o *ID* e a chave privada que o *Switch* fornece aos seus clientes e o ambiente em que será utilizada (teste ou produção).

- *CreateCard*: que recebe o *token* que é enviado pelo *Switch* para o cliente quando este lhe fornece os dados do cartão. Retorna os dados do cartão que o cliente pode guardar e um *token* que será utilizado para realizar os pagamentos.

- *CreatePayment*: que recebe o montante do pagamento, a moeda, e o cartão. Retorna o resultado do pagamento.

Nesta fase este módulo só suporta pagamentos em que o pagamento é realizado no momento da sua criação, não sendo possível reservar e mais tarde realizar a captura do pagamento ou libertar. A documentação deste módulo de *Parse* está também presente no anexo B.

Capítulo 6

Resultados

Neste capítulo são apresentados os resultados, utilizando, para isso, exemplos do fluxo de execução da aplicação. São apresentados também testes aos resultados nomeadamente o teste de usabilidade da aplicação e a análise da previsão da hora de entrega de uma refeição.

6.1 Fluxo de execução

6.1.1 Criar encomenda

A ação principal da aplicação é a criação da encomenda, essa ação pode ser realizada por um cliente mesmo antes de iniciar uma sessão.

A Figura 6.1 apresenta os ecrãs deste processo. O ecrã 1 é o ecrã de início de sessão, neste ecrã é possível escolher opções que permitem entrar na aplicação sem iniciar sessão, iniciar sessão (com endereço de correio eletrónico e password ou recorrendo ao *Facebook*) e registar. Se a escolha for iniciar sessão o cliente é encaminhado para o segundo ecrã. Se escolher registar aparecerá um ecrã para preencher os dados de registo e depois de registado será encaminhado também para o ecrã 2, se iniciar sessão (independentemente da forma) após a autenticação ser bem-sucedida o cliente é encaminhado também para o ecrã 2.

O ecrã 2 contém uma lista de restaurantes que se encontram perto da localização do cliente. O cliente neste ecrã pode pesquisar pelos restaurantes ou alterar a localização.

Figura 6.1: Criar encomenda

Quando o cliente escolhe um restaurante este é encaminhado para o ecrã 3, que apresenta os produtos disponíveis no restaurante, organizado por categorias. Ao selecionar um dos produtos aparecerá o ecrã 4 que permite ao cliente adicionar produtos a uma metáfora de carrinho compras. Neste ecrã pode definir a quantidade e acrescentar alguma observação para ser lida pelo estafeta.

O cliente pode adicionar vários produtos ao carrinho (mas só de um restaurante), e quando desejar pode consultar o seu carrinho, escolhendo a opção do mesmo na barra de opções no fundo dos ecrãs, abrindo o ecrã 5. Neste ecrã o cliente pode alterar a quantidade dos produtos e as observações, pode, também, remover os produtos. Pode, ainda, iniciar o *checkout* da encomenda.

Quando o cliente inicia o *checkout* terá de iniciar sessão se ainda não o tiver realizado e também terá de adicionar o seu cartão de crédito (se também ainda não o tiver realizado). Com a sessão iniciada aparecerá o ecrã 6, que permite o cliente indicar o local da entrega da refeição.

Após escolher o local da entrega é pedido ao cliente que complete algumas informações referentes à localização da entrega, nomeadamente o número de porta e o andar a ser introduzida no ecrã 7. Será então apresentado um ecrã com uma visão geral dos dados da encomenda, que inclui a previsão do tempo e do custo da entrega. Se o cliente confirmar, a encomenda é criada ficando a aplicação no ecrã 8 à espera de um estafeta para realizar a entrega.

6.1.2 Estado da encomenda

Após a criação da encomenda, o cliente pode acompanhar o estado da mesma. Para isso o cliente pode ver todas as suas encomendas (em execução e concluídas) e escolher uma para acompanhar o seu estado. Na Figura 6.2 são apresentados alguns ecrãs que permitem acompanhar

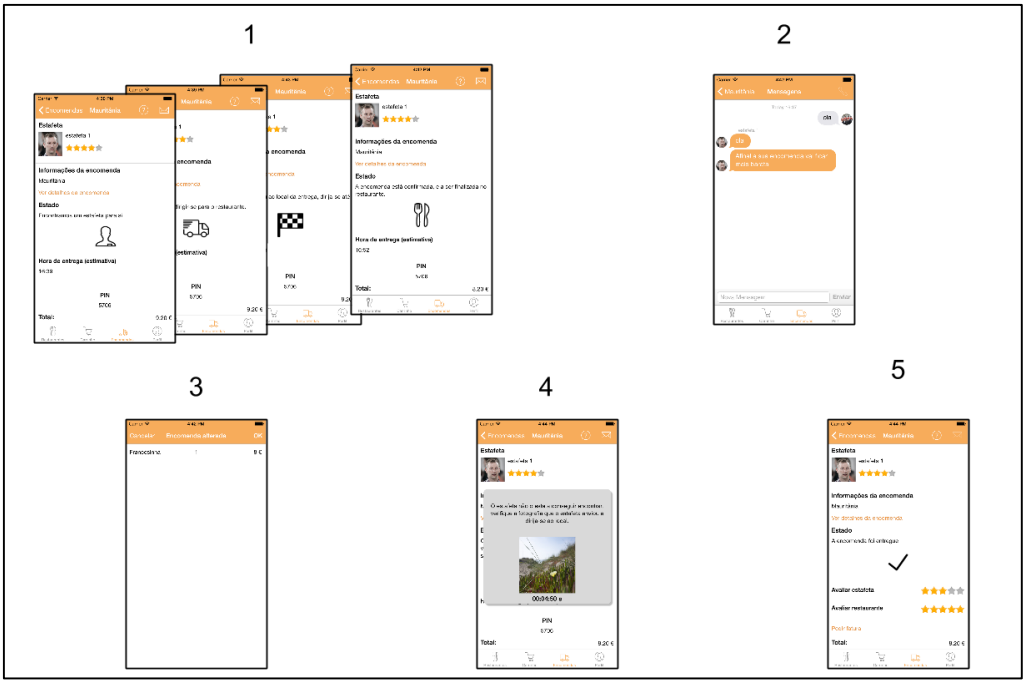

Figura 6.2: Acompanhamento do estado da encomenda

o estado. O conjunto de ecrãs 1 vai sendo atualizado com o desenrolar do estado da encomenda, sendo alterada a hora prevista de entrega e uma frase que indica o estado da mesma, juntamento com uma imagem ilustrativa desse estado.

O ecrã 2 é o ecrã que permite comunicar com o estafeta. O cliente pode aceder a ele tocando no botão do *envelope* na barra de navegação no ecrã do estado da encomenda (grupo 1 de ecrãs). Neste ecrã o cliente pode ligar ao estafeta e comunicar por um sistema de mensagens instantâneas.

O ecrã 3 aparece quando o cliente se encontra no ecrã de estado da encomenda (grupo 1 de ecrãs) e o estafeta altera o preço ou a quantidade de um produto. Este ecrã pede ao cliente para confirmar ou recusar a alteração que foi realizada à encomenda.

O ecrã 4 é um caso especial do ecrã de estado. Este ecrã aparece com este aspeto quando o estafeta chega ao local da entrega e não encontra o cliente. Na zona central é possível visualizar uma mensagem, que indica que o estafeta não está a conseguir encontrar o cliente, uma imagem que deve ser facilmente associável à localização do estafeta e um cronómetro que indica o tempo que o cliente tem para receber a encomenda.

Por ultimo o ecrã 5, que é também um caso especial do ecrã de estado, indica que a encomenda está terminada e foi entregue com sucesso. Nele o cliente pode avaliar o estafeta e o restaurante. Pode também pedir que lhe seja enviada uma fatura.

6.1.3 Estafeta

A aplicação quando é utilizada por um estafeta apresenta menos opções do que quando é utilizada por um cliente. A Figura 6.3 apresenta os ecrãs principais da aplicação quando é utilizada

por um estafeta. O ecrã 1 aparece após ser realizada a autenticação na aplicação. Este ecrã contém uma barra na parte inferior, onde o estafeta pode navegar para outros ecrãs da aplicação, nomeadamente ver as entregas que já realizou, o seu perfil e entrar em *modo de espera*. Sendo esta última a opção selecionada o estafeta pode ligar o *modo de espera*. Para isso só necessita de ligar o botão no centro do ecrã. Ao ligar ficará ativo no sistema, estando disponível para realizar entregas. Na aplicação, o ecrã ficará com o aspeto do ecrã 2 da figura desaparecendo a barra inferior impedindo que o estafeta navegue na aplicação.

Estando em *modo de espera* pode receber pedidos de entregas. Quando isso acontece é aberto um novo ecrã como o ecrã 3. Neste ecrã o estafeta consegue ver as informações gerais da

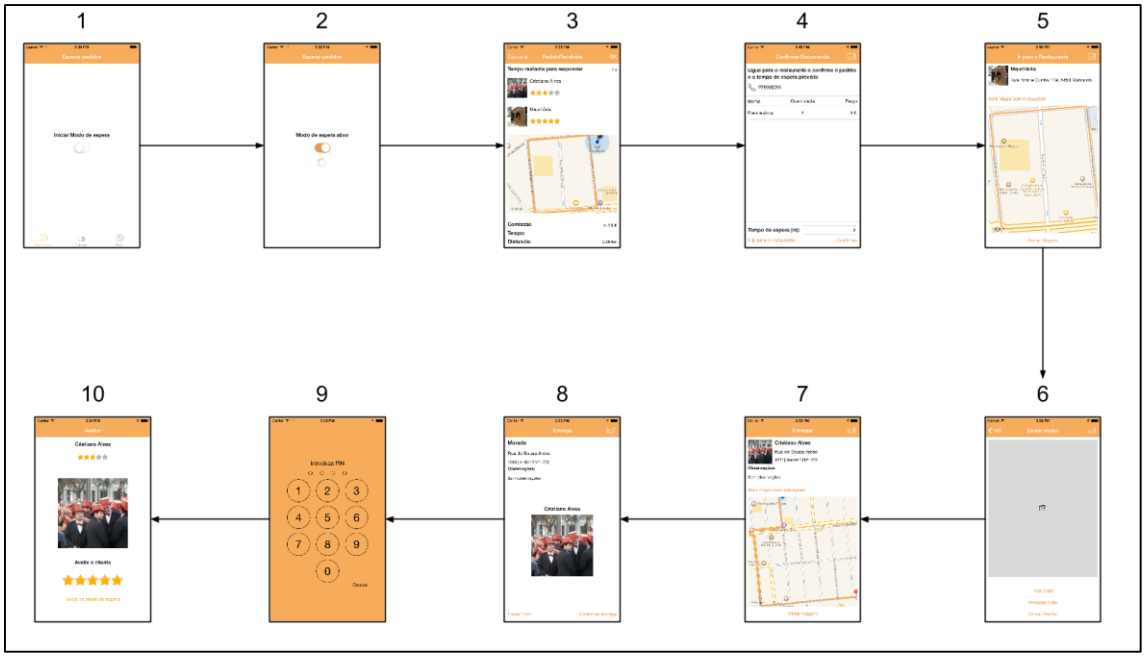

Figura 6.3: Ecrãs aplicação estafeta

encomenda e o tempo que ainda lhe resta para aceitar a encomenda. Se o estafeta recusar ou não responder o ecrã desaparecerá e voltara ao ecrã 2. Caso aceite, a aplicação irá navegar atá ao ecrã 4.

No ecrã 4 o estafeta tem a opção de iniciar viagem em direção ao restaurante ou fazer uma chamada para o restaurante de forma a confirmar a encomenda e perceber o tempo de preparação da encomenda. Se alterar a encomenda a aplicação irá navegar para um ecrã de espera, enquanto o cliente não aceitar a alteração. Caso não faça nenhuma alteração a aplicação navegará para o ecrã 5.

Neste ecrã 5 é apresentado um mapa com o percurso para o restaurante. O estafeta deve indicar o momento em que inicia a viagem e o momento em que termina. Se desejar pode abrir uma aplicação de navegação que possua no seu *iPhone*, escolhendo a opção para esse efeito.

Quando o estafeta indica que chegou ao restaurante aparecerá um ecrã para realizar uma nova confirmação da encomenda. De seguida é pedido um comprovativo de compra, devendo ser utilizada uma fotografia da fatura para fazer essa comprovação. O ecrã 6 tem esse objetivo,

permitindo escolher uma fotografia que o estafeta possua no seu *iPhone* ou fotografar o comprovativo, utilizando a própria aplicação.

Após o estafeta enviar o comprovativo poderá iniciar a viagem até ao local da entrega, aparecendo o ecrã número 7, que apresenta funções muito semelhantes ao ecrã número 5. Porém este apresenta instruções até ao local da entrega.

Chegando ao local da entrega o estafeta deve indicá-lo, aparecendo o ecrã número 8. Se porventura o estafeta não estiver a conseguir encontrar o cliente, poderá enviar uma fotografia que represente bem a sua localização. Caso contrário pode confirmar logo a encomenda.

Quando o estafeta escolhe a opção de confirmar a encomenda aparecerá o ecrã 9 onde deve ser introduzido o PIN que é fornecido ao cliente na criação da encomenda.

Uma vez confirmada a encomenda o estafeta pode avaliar o cliente, utilizando para isso o ecrã número 10.

6.2 Testes de usabilidade

De forma a avaliar a aplicação foi criado um conjunto de testes de usabilidade. Um teste de usabilidade tem como objetivo avaliar um produto em termos da efetividade, eficiência e satisfação na sua utilização por parte dos utilizadores. Para isso deve ser escolhido um grupo de utilizadores que representem os segmentos de utilizadores da aplicação quando esta estiver em fase de produção.

Dessa forma criou-se um pequeno grupo de utilizadores a quem foi pedido para realizar um conjunto de tarefas na aplicação, e durante a realização das mesmas foi analisado o tempo que o utilizador demorava a encontrar o ecrã/opção para realizar a tarefa.

Os testes foram divididos em duas partes: o teste da aplicação como cliente, e o teste da aplicação como estafeta. O enunciado dos testes encontra-se no anexo E. Algumas das tarefas foram adaptadas, dependendo do utilizador em questão, nomeadamente a tarefa referente à escolha da localização da entrega.

A realização de cada teste começou com uma breve apresentação da aplicação. Os objetivos da mesma e as suas funcionalidades foram apresentados oralmente sem utilizar qualquer imagem relativa à aplicação. Após isso, foi fornecido um *iPhone* com a aplicação aberta e foi indicada a primeira tarefa que o utilizador teria de realizar. O tempo que o mesmo levou a concluí-la foi então medido e registado, passando-se para a próxima tarefa e repetindo este processo.

Durante a realização dos testes foram também recolhidas impressões e opiniões dos utilizadores de forma a perceber quais os pontos onde a aplicação poderia ser melhorada.

O número de utilizadores que realizaram os testes de usabilidade foi reduzido, apresentando idades compreendidas entre os 19 e os 33 anos, sendo 33% do sexo feminino.

De forma a obter uma análise global foi criado o gráfico apresentado na Figura 6.4. Este gráfico contém a distribuição da frequência do tempo necessário para realizar a tarefa. Foram criados intervalos – <5, [6 – 10], [11 – 15], [16 – 20] e >20, em segundos – de forma a obter uma melhor análise da distribuição.

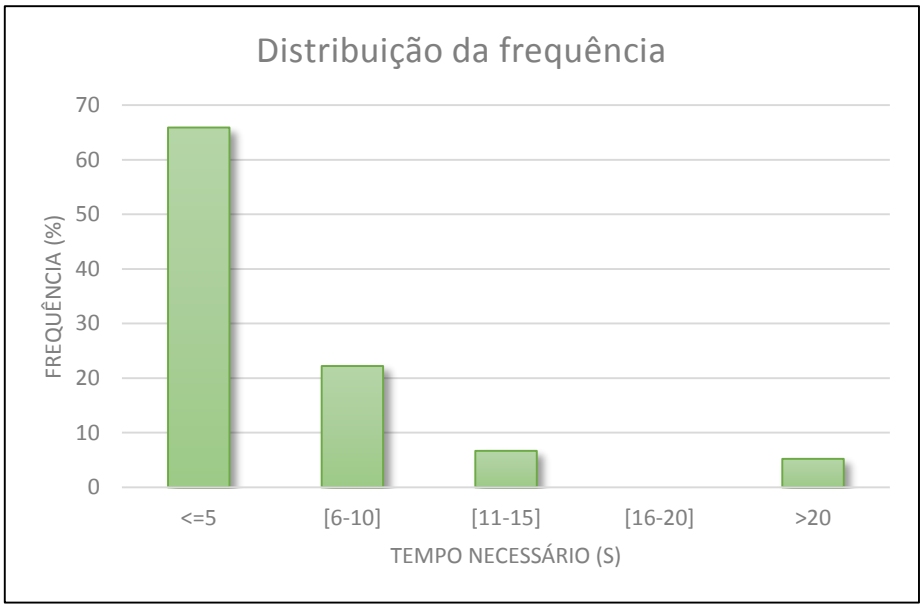

Figura 6.4: Distribuição da frequência de tempo necessário para realizar tarefa

Os resultados demonstram que 65% das tarefas realizadas pelos utilizadores durante os testes foram realizadas em menos de 5 segundos, já cerca de 22% das tarefas foram realizadas entre 6 a 10 segundos e apenas 6% entre 11 e 15 segundos. Embora o número de tarefas concluídas no intervalo de 16 segundos a 20 seja zero existem 5% em que foi necessário um tempo de conclusão superior a 20 segundos. No anexo E é apresentado um conjunto de dados estatísticos em relação a todos os testes.

De forma a compreender melhor estes resultados será agora analisado individualmente cada teste de forma a perceber onde se encontram os piores resultados e tentar compreender o porquê desses resultados de forma a melhorá-los no futuro.

6.2.1 Testes da aplicação Cliente

Durante a realização dos testes foi utilizada sempre a mesma ordem, começando sempre pelos testes à aplicação em modo cliente. O primeiro teste realizado consistia em efetuar o registo. Como o registo é um processo lento por implicar introdução de texto foi contabilizado apenas o tempo que o utilizador necessitou para encontrar o local de registo. Esta tarefa para uma percentagem considerável dos utilizadores revelou-se complicada sendo que só 22% dos utilizadores conseguiu encontrar a opção de registo num tempo igual ou inferior a 5 segundos. Já

a percentagem de utilizadores que demoraram mais de 10 segundos ficou-se nos 33% sendo a média de tempo necessário 11,5 segundos. A Figura 6.5 apresenta o ecrã onde se encontra o botão de registo. Os utilizadores criticaram a sua posição afirmando que se encontrava um pouco escondido. Afirmaram também que a melhor posição seria junto aos restantes botões do ecrã.

Após o registo, o utilizador era confrontado com a tarefa de terminar a sessão na aplicação.

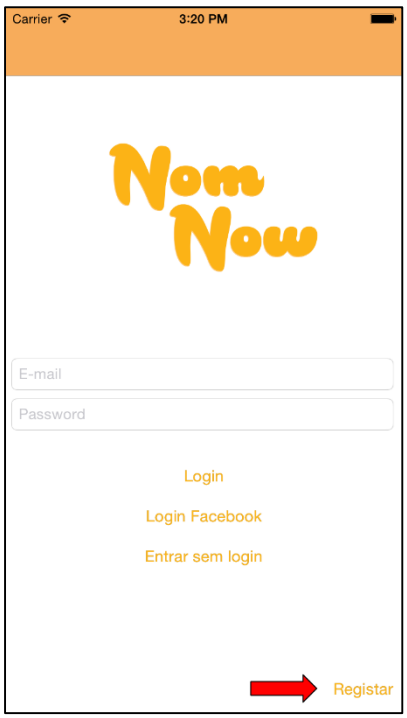

Figura 6.5: Posição do botão de registo

A aplicação encontrava-se num ecrã que apresentava uma lista de restaurante e o utilizador teria de navegar até ao ecrã de perfil para terminar sessão. Estes passos não eram indicados, tendo o utilizador de navegar pela aplicação de forma a encontrar a opção para terminar sessão. Quase 67% dos utilizadores conseguiu encontrar rapidamente a opção demorando um tempo igual ou inferior a 5 segundos. Apenas 11% dos utilizadores demorou mais que 10 segundos. Nesta tarefa os utilizadores não apresentaram críticas afirmando até que era intuitiva a localização da opção de terminar a sessão.

A tarefa que se seguiu consistiu em iniciar sessão com a conta que anteriormente teria sido criada. O utilizador encontrava-se no ecrã visível na Figura 6.5 onde teriam de preencher os campos com o endereço de correio eletrónico e a sua palavra passe. Embora 66% dos utilizadores tenham percebido o que teriam de realizar para iniciar a sessão (a contagem parava quando o utilizador começasse a escrever o endereço de correio eletrónico) num tempo igual ou inferior a 5 segundo e 100% tenham demorado menos de 10 segundos, existiram alguns utilizadores que apresentaram algumas dúvidas e apresentaram algumas sugestões. O ecrã encontra-se dividido em quatro partes: o logotipo, as caixas de texto para o utilizador indicar os dados de início de sessão, a secção de botões de início de sessão e o botão de registo. Alguns utilizadores quando confrontados com a indicação que teriam de iniciar sessão tiveram como primeira reação tocar no

botão de início de sessão sem preencher os dados, o que leva ao aparecimento de uma mensagem de erro que indica que teriam de preencher os dados e desta forma concluir a tarefa. Alguns utilizadores indicaram que este ecrã poderia só apresentar apenas botões e num segundo ecrã é que seria apresentado o local para a introdução dos dados.

A tarefa seguinte é a tarefa que incluiu mais passos até ser concluída, mas sem necessitar de introdução de texto. Desta forma foi contabilizado o tempo até a sua conclusão. Esta tarefa consistiu em adicionar um produto ao carrinho, sendo indicado ao utilizador o restaurante que deveria selecionar e o produto que deveria adicionar ao carrinho. Como era uma tarefa que envolve mais operações o tempo de conclusão da mesma foi também superior às restantes, apresentando uma média de 12,5 segundos em que apenas 11% dos utilizadores consegui concluir a tarefa num tempo igual ou inferior a 5 segundos e 44% dos utilizadores demorou mais de 11 segundos. Embora o tempo da realização da tarefa seja elevado não foi considerado um mau resultado pois era uma tarefa mais longa. No entanto houve alguns utilizadores que apresentaram

Figura 6.6: Adicionar produto ao carrinho

dificuldades ao adicionar o produto ao carrinho. A Figura 6.6 apresenta o ecrã onde é realizada a operação de adicionar um produto ao carrinho, alguns utilizadores tocavam no título do ecrã - "Adicionar Produto" – erradamente, em vez de tocarem no botão com o caráter "+". Os utilizadores que apresentaram esta confusão referiram que a visualização do caráter "+" era ofuscado pelo título e sugeriram a troca do caráter por uma imagem que relacionasse o ato de adicionar com o carrinho.

A tarefa número 5 consistiu na alteração da quantidade de um produto já adicionado ao carrinho, onde o utilizador teria de navegar até ao ecrã do carrinho, selecionar o produto a alterar,

alterar a quantidade e guardar a alteração. De modo semelhante com a tarefa anterior, esta tarefa também implicava uma navegação na aplicação. Aqui 88% dos utilizadores concluiu a tarefa em tempo igual ou inferior a 10 segundos. Esta tarefa não apresentou críticas dos utilizadores, considerando que o fluxo que executaram era lógico.

A tarefa seguinte consistiu em avançar no processo da encomenda, que necessita apenas de tocar num botão. Revelou-se simples já que 78% não necessitou de mais de 5 segundos para concluir a tarefa. Em média cada utilizador necessitou de 8 segundos para essa conclusão.

A tarefa número 7 consistiu em escolher o local de entrega da encomenda. No enunciado indicou-se que o utilizador deverá introduzir a localização da empresa *Glazed Solutions*. Porém em alguns casos foi utilizada outra localização. A localização foi sempre uma localização que o utilizador conhecesse, já que o objetivo não era medir o tempo que o utilizador demorava a marcar a localização da entrega mas sim medir o tempo que o utilizador demorava a perceber como funcionava o método de escolha da localização da entrega. A Figura 6.7 apresenta o ecrã da

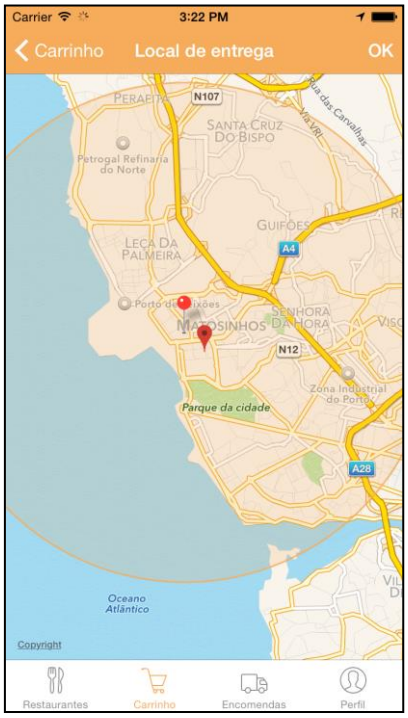

Figura 6.7: Escolha da localização da entrega

escolha do local da entrega. Na imagem é possível visualizar um "alfinete" que ao tocar irá indicar que representa a localização do restaurante, um seletor no centro do mapa que indica a localização da entrega e um círculo em laranja que indica a área admissível de entrega. Para o utilizador escolher a localização deve deslocar o mapa e não o seletor, este estará sempre presente no centro do mapa. Esse pormenor causou alguma confusão nos utilizadores levando a que o tempo médio que um utilizador demorou a perceber o funcionamento do mapa fosse de quase 12 segundos. Em 44% dos casos demorou-se mais de 10 segundos a perceber o funcionamento do mapa. Os

utilizadores relataram no final que achavam que era o seletor que deveriam mover, mas admitiram, também, que não era difícil de depois perceber o mecanismo implementado.

A última tarefa do teste da aplicação em modo cliente consistiu em confirmar a encomenda de modo a ficar à espera do estafeta. A realização desta tarefa era semelhante a outras que o cliente já tinha realizado, pelo que foi concluída muito rapidamente, com uma média de 2 segundos.

6.2.2 Testes da aplicação Estafeta

Os testes realizados na aplicação em modo estafeta apresentaram melhores resultados comparativamente aos testes na aplicação em modo cliente. Isto acontece por dois motivos: por um lado os utilizadores já se encontravam familiarizados com a aplicação, e por outro as tarefas eram mais simples. Na aplicação em modo cliente os utilizadores demoraram em média 8 segundos a realizar cada tarefa, já na aplicação estafeta este tempo foi reduzido para menos de metade passando para os 3,9 segundos.

A primeira tarefa era iniciar sessão como estafeta, semelhante aos testes da aplicação em modo de cliente. Com isto todos os utilizadores conseguiram concluir a tarefa com um tempo igual ou inferior a 5 segundos.

A segunda tarefa pedia ao utilizador que consultasse a lista de entregas que o estafeta autenticado já tinha realizado. Esta tarefa implica tocar numa opção na *tab bar* na parte inferior do ecrã da aplicação. Em que 89% dos casos os utilizadores conseguiram concluir a tarefa com um tempo igual ou inferior a 5 segundos, com uma média de 3,4 segundos.

De seguida o utilizador deveria avaliar um cliente para quem o estafeta autenticado já tivesse realizado uma entrega. Esta tarefa implica que o utilizador abra uma entrega já concluída e depois escolha a opção para classificar o cliente. Esta tarefa causou alguma confusão aos utilizadores. A maioria percebeu imediatamente que teria de abrir uma entrega para classificar o cliente, porém erravam na escolha da opção da classificação. A Figura 6.8 apresenta marcado o erro praticado pelos utilizadores ao classificar o cliente. A primeira intuição dos utilizadores é tocar nas estrelas que aprecem junto ao cliente, porém essas estrelas apresentam a classificação média do cliente em questão, e para classificar é necessário selecionar o botão “Classificar Cliente”. Mesmo com esta confusão 44% dos utilizadores conclui a tarefa com um tempo igual ou inferior a 5 segundos sendo a média de tempo necessária de 11 segundos. Os utilizadores que sentiram mais dificuldades sugeriram utilizar as estrelas para classificar ou apresentar cores diferentes nas estrelas e no botão de forma a existir um contraste e ser mais percetível para o utilizador o que representa cada componente, as estrelas e o botão.

Figura 6.8: Classificar cliente

A quarta tarefa consistiu em consultar os detalhes de uma encomenda já realizada. Os detalhes da encomenda são os produtos que foram entregues na realização da entrega, com as respetivas quantidades e preços. Essa opção encontra-se nas informações da entrega. Esta tarefa foi realizada facilmente pelos utilizadores com uma média de 3 segundos.

A quinta tarefa consistiu em ligar o modo de espera do estafeta. Para isso é necessário escolher a opção presente na *tab bar*, na parte inferior do ecrã, para esse efeito, e nesse ecrã selecionar o botão para ligar o modo de espera. Os utilizadores não apresentaram grandes dificuldades em cumprir essa tarefa cumprindo-a com uma média de 3 segundos, onde 89% dos utilizadores demoraram até 5 segundos a realizar a mesma.

A sexta tarefa aparece no seguimento da quinta, é proposto ao utilizador que desligue o modo de espera. Para isso é apenas necessário selecionar o mesmo botão que foi selecionado para ligar esse modo. Novamente foi uma tarefa simples para os utilizadores, apresentando um tempo médio de conclusão de 1,6 segundos.

A última tarefa é uma tarefa que os utilizadores já tinham realizado no teste da aplicação em modo de cliente, terminar a sessão. Devido a isso os utilizadores completaram a tarefa rapidamente com uma média de 1,4 segundos.

A maioria dos testes realizados apresentaram bons resultados já que os utilizadores conseguiram realizar as tarefas sugeridas em poucos segundos. Por outro lado, ficou claro que existem alguns pontos que criaram confusão aos utilizadores. Todavia, é essa a finalidade dos testes de usabilidade – descobrir problemas de usabilidade. Os resultados destes testes, que incluem o tempo de conclusão das tarefas, mas também as críticas dos utilizadores, serão uma

ajuda para o trabalho futuro de criação do produto, conseguindo-se assim melhorar a usabilidade e melhor adaptando a aplicação aos seus utilizadores.

6.3 Avaliação de resultados

De forma a avaliar o funcionamento da aplicação, nomeadamente a previsão de hora de entrega, foi realizado um conjunto de simulações de encomendas e respetivas entregas.

Estas simulações foram realizadas em ambiente real, em que um estafeta recebia a encomenda e deslocava-se a um restaurante, fazia o pedido e quando lhe fosse entregue a refeição ia ao encontro do cliente para entregar a refeição.

A simulação foi realizada durante vários dias utilizando sempre o mesmo percurso, o mesmo estafeta e o mesmo restaurante. O objetivo foi analisar se a previsão de tempo apresentava valores próximos dos reais e se se realizava uma boa aprendizagem, melhorando os resultados à medida que se iam realizando as simulações.

Como foram realizadas simulações em ambiente real o número de simulações foi reduzido sendo realizadas apenas sete simulações. Nas simulações o estafeta deslocava-se a pé tendo de realizar um percurso pequeno até à localização do restaurante e depois até à localização da entrega. Como já foi apresentado no capítulo relativo à implementação, a previsão de tempo é apresentada durante a própria encomenda, sendo atualizada quando o estado da encomenda é alterado. Assim sendo, foram recolhidos os dados da previsão em três estados diferentes:

- Quando o estafeta indica que se está a dirigir para o restaurante;
- Quando o estafeta indica que chegou ao restaurante;
- Quando o estafeta indica que se está a dirigir para a localização da entrega.

Com os resultados recolhidos foram criados dois gráficos de forma analisar melhor os resultados. O primeiro gráfico presente na Figura 6.9 apresenta o valor do erro da previsão do tempo de espera. É possível verificar que o erro foi diminuído com o número de simulações realizadas. Porém existiram algumas exceções como é o caso do erro da previsão realizada quando o estafeta sai do restaurante. Nesse caso, o valor do erro apresenta grandes oscilações. Essas oscilações acontecem devido à distância que o estafeta percorre entre o restaurante e a localização da entrega ser muito curta que leva a que o estafeta realize esse percurso em poucos minutos de modo a que quando se verifica uma pequena diferença (1 ou 2 minutos) isso pode levar a erros na ordem dos 50%. De forma a ultrapassar o problema desta métrica de avaliação foi criando um outro gráfico que apresenta a diferença entre a previsão e a hora real de entrega, em valor absoluto, ou seja em minutos.

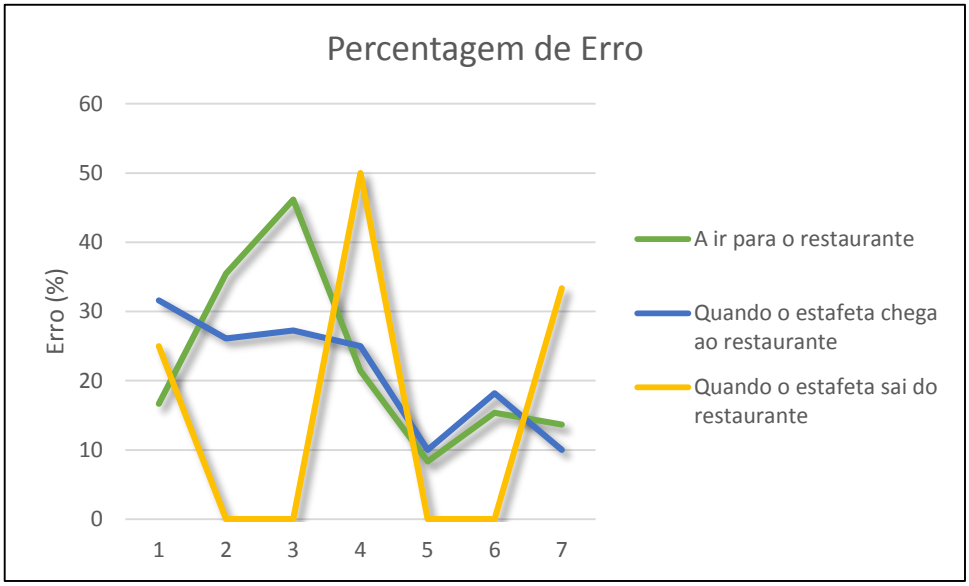

Figura 6.9: Percentagem de erro

Esse gráfico, presente na Figura 6.10, apresenta uma visão de mais fácil interpretação das simulações realizadas. É possível observar que a primeira previsão em cada simulação apresenta, em geral, uma maior diferença em relação às restantes previsões. Como o resultado é recalculado nessas três situações diferentes, as previsões finais têm intervalos de tempo menores para estimar, conseguindo assim uma melhor previsão.

É de salientar que na primeira simulação, a primeira previsão é calculada utilizando uma velocidade por defeito para o estafeta (60 Km/h) e o tempo de espera no restaurante é de 10 minutos por produto. Na Figura 6.10 é possível perceber que na primeira simulação a diferença

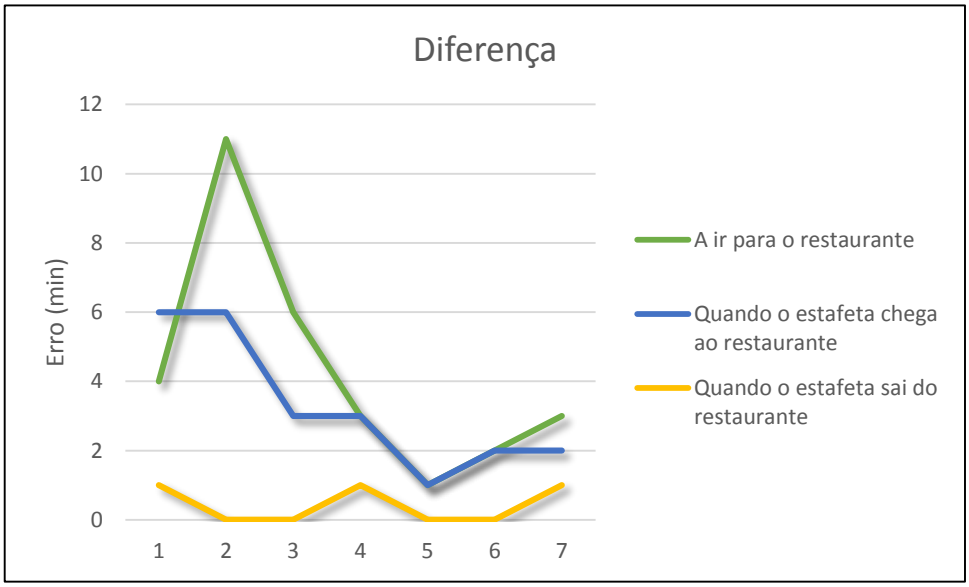

Figura 6.10: Diferença entre a previsão e a espera real

entre a previsão e a espera total, no momento que o estafeta sai do restaurante é de apenas uma minuto, ao contrário da primeira previsão onde a diferença é de quatro minutos. O motivo da

melhoria da previsão na mesma simulação está relacionado com o cálculo que é realizado. Na terceira previsão já são utilizados dados estatísticos do percurso que o estafeta realizou entre a sua localização e o restaurante, melhorando assim a previsão.

É possível observar, também, que os resultados da previsão foram melhorando com o número de simulações realizadas passando os erros a serem bastante baixos. Nas últimas três simulações o erro maior foi de três minutos, na primeira previsão.

Embora o número de simulações tenha sido baixo foi possível observar que existiu uma aprendizagem por parte da aplicação na previsão do tempo de espera.

Capítulo 7

Conclusão

Este último capítulo irá analisar o trabalho desenvolvido, apresentando um resumo sendo também analisadas as decisões tomadas durante o desenvolvimento da aplicação e indicando as vantagens e as desvantagens dessas escolhas, apresentando as devidas conclusões. No final são apresentadas as linhas futuras do desenvolvimento do trabalho realizado. Essas linhas seguirão as conclusões obtidas anteriormente, com vista a melhorar e a corrigir erros que aconteceram no desenvolvimento da aplicação.

7.1 Resumo do trabalho desenvolvido

Durante a realização desta dissertação foi desenvolvido um serviço de entrega de refeições utilizando o modelo de negócio da *share economy*. Para o seu desenvolvimento foi necessário numa primeira fase realizar um estudo aprofundado do modelo de negócio, de forma a perceber o seu funcionamento, o tipo de serviços que poderia abranger e que outros conhecimentos seriam necessários para o desenvolvimento do serviço com este modelo de negócio.

Após esta análise, foram definidos os requisitos do serviço e criada a arquitetura do mesmo. A arquitetura pode ser dividida em dois grandes componentes, uma aplicação móvel e o *backend* do serviço.

No *backend* foi implementada grande parte da lógica de negócio do serviço, e foram também armazenados os dados do mesmo. A sua implementação recorreu a uma tecnologia *noBacked*, o *Parse*. Embora o *Parse* simplifique o desenvolvimento de um *backend*, fornecendo as interações básicas como objetos, foi necessário implementar funções – *cloud functions* – e *triggers* para garantir o bom funcionamento do serviço ao nível da lógica de negócio.

O outro grande componente é a aplicação móvel que foi desenvolvida para o sistema operativo móvel da *Apple – iOS*, utilizando *Objective-C* para a sua implementação. Esta aplicação foi dividida em três componentes, a componente do estafeta, a componente do cliente e a componente comum. A aplicação, dependendo do tipo de utilizador (estafeta ou cliente),

comporta-se de forma diferente. A aplicação permite ao cliente encomendar refeições e acompanhar o estado da entrega das suas encomendas. Já ao estafeta permite receber pedidos de realização de entregas, aceitar ou recusar as mesmas e atualizar o seu estado, caso as aceite. Na aplicação foram, também, implementados mecanismos para ajudar na resolução de conflitos entre utilizadores e mecanismos que fomentam a confiança entre utilizadores e no serviço.

Foi também criado um SDK simples para utilizar o serviço de pagamentos *Switch Payments*, desenvolvido em *Objective-C* para facilitar a integração da componente cliente. Para juntar a este SDK foi criado um *Cloud Module* de *Parse* em *JavaScript* para integrar o *backend* com o *Switch Payments*.

7.2 Avaliação das escolhas tecnológicas

O desenvolvimento desta dissertação obrigou a realizar escolhas tecnológicas que em geral se revelaram escolhas positivas.

Uma das grandes escolhas foi o sistema operativo móvel. O sistema operativo móvel escolhido foi o *iOS* em detrimento de outros sistemas operativos e de *frameworks* que permitem o desenvolvimento de aplicações para *multi* sistemas operativos. Esta escolha revelou-se uma escolha interessante do ponto de vista da aprendizagem, também da rapidez de desenvolvimento de aplicações e da interação com a tecnologia escolhida para realizar o *backend* da aplicação. Dentro da escolha do *iOS* existia a escolha da linguagem a ser utilizada tendo-se optado pelo *Objective-C* em detrimento do *Swift*. Esta escolha justificou-se pela maturidade do *Objective-C* face ao *Swift.* Como não foi realizado um teste entre as duas linguagens não é possível avaliar a qualidade da escolha, porém o desenvolvimento com *Objective-C* não foi um problema.

A escolha do *Parse* para ser utilizado para o *backend* foi sugerida pela empresa de acolhimento. Tendo em conta o objetivo da dissertação que passava por desenvolver um protótipo (e prova de conceito), esta tecnologia foi de facto uma boa escolha. Esta conclusão deve-se ao facto do *Parse* permitir um desenvolvimento muito rápido, seja pelo facto de possuir um SDK para *iOS* (e outras tecnologias) que facilita o desenvolvimento da comunicação entre estes dois componentes, seja por criar automaticamente métodos de manipulação de objetos ao contrário do desenvolvimento de uma API convencional em que se teria de desenvolver um conjunto de endereços de acesso para cada entidade. O *Parse* possui também uma alta disponibilidade e para pequenos projetos é gratuito. E é neste ponto que aparecem os problemas do Parse. Embora seja gratuito está limitado a 30 pedidos por segundo, e quando o serviço necessita de mais pedidos é necessário subscrever um plano do serviço, passando a ser um serviço pago.

Nesta aplicação por cada encomenda ativa no sistema são realizados dois pedidos ao *Parse* em cada quatro segundos, o serviço possui sempre um *background job* que executa um pedido por segundo. E quando existe um *background job* o limite de pedidos desce para 20. Com estas restrições o serviço possui capacidade para funcionar corretamente com um total de 38

encomendas em simultâneo. A Figura 7.1 apresenta o crescimento do número de pedidos relacionando-o com as encomendas.

A aplicação suportar 38 encomendas numa primeira fase era um valor aceitável, mas podem existir muitos outros pedidos a acontecer no serviço.

Um outro tipo de pedidos acontece no momento em que um estafeta se encontra à espera de uma encomenda. Nessa situação cada estafeta realiza um pedido em cada dois segundos. Com 19 estafetas nessa situação o limite de 20 pedidos por segundo era atingido.

Figura 7.1: Número de pedidos por segundo dependendo do número de encomendas

Se se conjugarem estes dois parâmetros é possível analisar que por cada estafeta que esteja em modo de espera, o número de encomendas ativas máximo diminui em duas unidades. Assim sendo, estando 10 estafetas em modo de espera só seria possível funcionar de forma correta com 18 encomendas em simultâneo.

Como já foi referido é possível passar para um plano pago do *Parse*, porém estes planos apresentam valores bastante elevados. Para aumentar o limite em 10 pedidos por segundo o custo é de $100 por mês.

Concluindo, o *Parse* é útil para desenvolver aplicações pequenas, não necessitando de se atualizarem em tempo real. Também é útil para pequenos protótipos ou provas de conceito. Para todas as aplicações que necessitem de muitas trocas de informação o *Parse* torna-se uma solução bastante dispendiosa.

A outra escolha tecnológica foi o sistema de pagamentos, tendo sido escolhido o *Switch* por sugestão da empresa de acolhimento. Revelou-se uma escolha interessante por ser simples de ser utilizado e por dar a possibilidade de desenvolver SDK's para futuros utilizadores poderem ter uma melhor experiência com o serviço.

7.3 Avaliação do trabalho desenvolvido

O grande objetivo da dissertação passava por desenvolver uma aplicação móvel que permitisse realizar encomendas de refeições, e as respetivas entregas, utilizando o modelo de negócio da *share economy*. Esse grande objetivo foi comprido estando a aplicação desenvolvida a cumprir com os requisitos apresentados na Tabela 4.1.

A aplicação foi também alvo de testes de usabilidade que abordaram algumas das narrativas de utilização presentes na Tabela 4.2. Esses testes revelaram que os utilizadores se sentiram bastante à vontade ao utilizar a aplicação, existindo apenas alguns pormenores de devem ser melhorados para criar uma melhor experiência de utilização aos seus utilizadores.

Um outro objetivo passava por encontrar métodos que criassem confiança nos utilizadores ao utilizar a aplicação. Foi criado, então, um sistema de classificação de utilizadores e restaurantes, que permite a clientes avaliarem estafetas e restaurantes; e estafetas avaliarem clientes. Com isto é apresentada uma classificação média do restaurante ao cliente no momento da encomenda. E no momento da escolha do estafeta, que realizará a entrega da encomenda do cliente, o sistema utilizará a classificação média dos estafetas para escolher o melhor estafeta disponível (utilizando outros parâmetros para essa escolha). Já o estafeta consegue visualizar a classificação do cliente podendo aceitar ou recusar.

Esta dissertação tinha, também, como objetivo o desenvolvimento e implementação de métodos de resolução de conflitos. Na secção 5.3.5 foram apresentados vários métodos que fornecem dados que ajudam os gestores de sistema a resolver conflitos entre utilizadores.

Avaliando a dissertação do que diz respeito ao cumprimento dos objetivos creio que foi bem sucedida. Foram criadas soluções para cumprir os grandes objetivos da dissertação, como foi apresentado nesta secção.

Já do ponto de vista da resolução do problema apresentado na secção 1.2, o problema foi parcialmente resolvido. Foi criada uma aplicação que segue o modelo de negócio da *share economy*, porém existe um problema que a *share economy* apresenta que não foi muito abordado nesta dissertação – os problemas legais. Embora existam algumas narrativas de utilização que foram desenvolvidas de forma a ultrapassar alguns dos problemas legais que uma aplicação deste género possa vir a apresentar, como por exemplo a possibilidade de um cliente pedir uma fatura, esse problema foi deixado um pouco em segundo plano. Foi considerado que este problema deve ser resolvido ao nível do modelo de negócio, fugindo do âmbito da prova de conceito realizada. Todavia será algo muito importante a ser analisado com atenção no futuro, fazendo parte do trabalho futuro que será exposto na secção seguinte.

7.4 Desenvolvimento futuro

Os objetivos da dissertação foram compridos porém, como já foi referido existem escolhas que se percebeu que para um produto em produção não são viáveis.

A escolha que se tornou mais problemática para o projeto foi o *Parse*, como já foi referido anteriormente. O *Parse* tem limitações que não podem existir quando o serviço estiver em produção. Tendo isso em conta, o *backend* do serviço terá de ser reformulado, começando por abandonar o *Parse* e optar por uma outra tecnologia para o desenvolver. Essa tecnologia não deve ter as limitações apresentadas pelo *Parse*, podendo ser útil suportar a utilização de *websockets* e processamento em segundo plano.

Escolhendo esta nova tecnologia seria também importante implementar testes unitários ao *backend*. Algo que não foi possível devido às limitações do *Parse*, que não apresenta suporte para a implementação de testes.

Com a alteração da tecnologia do *backend* será necessário adaptar a aplicação para funcionar com o novo *backend*. A aplicação foi desenvolvida por componentes, levando com que a alteração do *backend* só afete o componente responsável pela comunicação com o *backend*.

A alteração do *backend* disponibiliza também outras opções para a aplicação que anteriormente não eram viáveis com a utilização do *Parse*. Uma dessas possíveis alterações passará por implementar *websockets* eliminando assim os *long polling* que existem atualmente na aplicação e tornando-a mais eficiente.

Também deverá investir-se mais no *design* da aplicação. Primeiro é importante criar uma aparência mais apelativa recorrendo possivelmente a um profissional na área de *design*. O *design* deverá ter em atenção as várias dimensões de ecrãs possíveis, apresentando resultados semelhantes em cada uma delas.

Ao nível dos algoritmos utilizados, o da previsão da hora de entrega necessita de uma reformulação de forma a se utilizarem mais dados importantes. Embora nos testes realizados os resultados tenham sido satisfatórios será importante dotar o algoritmo com mais dados que o possam melhorar, como a intensidade do trânsito no percurso do estafeta e o veículo de transporte que o mesmo irá utilizar. Estes novos dados poderão prever e apresentar melhores resultados, mesmo em situações anormais como é o caso da existência de um grande congestionamento de trânsito no percurso do estafeta, e a troca do meio de transporte.

Por fim, mas não menos importante, será necessário analisar a viabilidade legal da solução desenvolvida, adaptando-a à realidade portuguesa.

Referências

[14a] *Cancellations*. URL: http://uberpeople.net/threads/cancellations.6487i2014a.

[14b] *Driver cancel versus rider cancel*. URL: http://uberpeople.net/threads/driver-cancel-versus-rider-cancel.4830/i2014b.

[14c] *Robbed of a Cancellation Fee*. URL: http://uberpeople.net/threads/robbed-of-a-cancellation-fee.1560/i2014c.

[Appl14] Apple *About the iOS Technologies*. URL: https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html#//apple_ref/doc/uid/TP40007898-CH1-SW1i2014.

[Banc09] Banco de Portugal *Glossário*. URL: http://www.bportugal.pt/pt-pt/Glossarios/Paginas/Glossario.aspx?letter=Ai2009.

[Banc13] Banco de Portugal *SEPA - Área Única de Pagamentos em Euros*. URL: https://www.bportugal.pt/pt-pt/pagamentos/sepa/Paginas/inicio.aspx (acedido a última vez em November 11, 2014)i2013.

[Banc15] Bancaleiro Cláudia Uber proibida de operar em Portugal. In: *Público* (2015).

[Bbc14a] BBC *Uber taxi app suspended in Spain*. URL: http://www.bbc.com/news/business-30395093 (acedido a última vez em December 21, 2014)i2014a.

[Bbc14b] BBC *Uber banned in Delhi over taxi driver "rape."* URL: http://www.bbc.com/news/world-asia-india-30374070 (acedido a última vez em December 21, 2014)i2014b.

[Bird14] Birdsall Michelle Carsharing in a Sharing Economy. In: *ITE Journal* (2014), Nr. April, pp. 37–40.

[Böck13] Böckmann Marco *The Shared Economy: It is time to start caring about sharing; value creating factors in the shared economy*, University of Twente, 2013.

[Boku14] Boku *Why Boku*. URL: http://www.boku.com/why-boku/ (acedido a última vez em November 15, 2014)i2014.

[Bris15] Bristowe John *What is a Hybrid Mobile App?*. URL: http://developer.telerik.com/featured/what-is-a-hybrid-mobile-app/i2015.

[Buca13] Bucanek James *Learn IOS 7 App Development*. 1. ed. : Apress, 2013.

[Budi13] BUDIU RALUCA *Mobile: Native Apps, Web Apps, and Hybrid Apps*. URL: http://www.nngroup.com/articles/mobile-native-apps/i2013.

[Chou04] Chou, Y., Lee, C., and Chung J. Understanding M-commerce payment systems through the analytic hierarchy process. In: *Journal of Business Research* vol. 57 (2004), pp. 1423–1430.

[Comm14] Commission European *Vice-President Michel Barnier welcomes major milestone for the internal payments market with the migration to SEPA (Single Euro Payments Area)*. URL: http://europa.eu/rapid/press-release_STATEMENT-14-246_en.htmi2014.

[Cott17] Cotteleer, Mark J. and Cotteleer, Christopher A. and Prochnow Andrew Cutting Checks: Challenges and Choices in B2B e-Payments. In: *Commun. ACM* vol. 50 (2017), Nr. 00001-0782, pp. 56–61.

[Cred13] CreditCards.com *How a credit card is processed*, 2013.

[DaGr07] Dai Xiaoling and Grundy John NetPay: An off-line, decentralized micro-payment system for thin-client applications. In: *Electronic Commerce Research and Applications* vol. 6 (2007), Nr. 1, pp. 91–101.

[Edwa14] Edwards Jim *The iPhone 6 Had Better Be Amazing And Cheap, Because Apple Is Losing The War To Android*. URL: http://www.businessinsider.com/iphone-v-android-market-share-2014-5 (acedido a última vez em November 11, 2014)i2014.

[Frie13] Friedman Thomas L. Welcome to the "Sharing Economy." In: *New York Times* vol. 162. New York (2013).

[Gamb88] Gambetta Diego Can We Trust Trust? In: *Trust: Making and Breaking Cooperative Relations* : Blackwell, 1988, pp. 213–237.

[Gero13] Geron Tomio *Airbnb And The Unstoppable Rise Of The Share Economy*. URL: http://www.forbes.com/sites/tomiogeron/2013/01/23/airbnb-and-the-unstoppable-rise-of-the-share-economy/ (acedido a última vez em December 20, 2014)i2013.

[Gonz04] González AG PayPal: the legal status of C2C payment systems. In: *Computer Law & Security Review* (2004), pp. 1–14.

[Goog14] Google *Goole Wallet - Available Countries*. URL: https://support.google.com/wallet/answer/2604797?hl=en-GB (acedido a última vez em December 30, 2014)i2014.

[Goog15] Google *Google Maps JavaScript API v3*. URL: https://developers.google.com/maps/web/i2015.

[Gree00] Green Charles H White Paper Trust and the Sharing Economy: A New Business Model. In: *Trusted Advisor* (2000).

[Gsmh14] GSM History *VINTAGE MOBILES*. URL: http://www.gsmhistory.com/vintage-mobiles/ (acedido a última vez em December 05, 2015)i2014.

[Guan03] Guan, Sheng-Uei and Hua Feng A Multi-Agent Architecture for Electronic Payment. In: *International Journal of Information Technology Decision Making Decision Making* vol. 02 (2003), pp. 497–522.

[Hegg13] Heggestuen John *One In Every 5 People In The World Own A Smartphone, One In Every 17 Own A Tablet*. URL: http://www.businessinsider.com/smartphone-and-tablet-penetration-2013-10 (acedido a última vez em November 11, 2014)i2013.

[Hehr14] Hehr Nick *Getting Started with noBackend*. URL: http://nobackend.org/2014/05/getting-started-with-noBackend.html (acedido a última vez em January 12, 2015)i2014.

[Heng14] Heng Christopher *How to Create a Cron Job (Scheduled Task) for Your Website or Blog*. URL: http://www.thesitewizard.com/general/set-cron-job.shtmli2014.

[Hsie02] Hsieh E-commerce payment systems: critical issues and management strategies. In: *Human Systems Management* vol. 20 (202AD), pp. 131–138.

[Inve14] Investopedia *Sharing Economy*. URL: http://www.investopedia.com/terms/s/sharing-economy.asp (acedido a última vez em November 01, 2014)i2014.

[Iyer14] Iyer Anand *How Modern Marketplaces Like Uber And Airbnb Build Trust To Achieve Liquidity*. URL: http://techcrunch.com/2014/03/04/how-modern-marketplaces-like-uber-and-airbnb-build-trust-to-achieve-liquidity/ (acedido a última vez em January 04, 2015)i2014.

[Jack11] Jack McGrath *Introduction to iOS Development: An Overview of Objective-C*. URL: http://www.technobuffalo.com/2011/03/27/introduction-to-ios-development-an-overview-of-objective-c/i2011.

[Jami11] Jamille *O iphone da apple iphone foi o primeiro celular touchscreen de sucesso*. URL: http://www.maniadecelular.com.br/207263/o-iphone-da-apple-iphone-foi-o-primeiro-celular-touchscreen-de-sucesso.html (acedido a última vez em January 05, 2014)i2011.

[Jews01] Jewson R. e-Payments: credit cards on the Interne. In: *White Paper* (2001), pp. 1–7.

[JøIB07] Jøsang Audun , Ismail Roslan and Boyd Colin A survey of trust and reputation systems for online service provision. In: *Decision Support Systems* vol. 43 (2007), Nr. 2, pp. 618–644.

[Jumi14] Jumio *Supported Countries*. URL: https://www.jumio.com/netverify/supported-countries/ (acedido a última vez em December 18, 2014)i2014.

[Just12] Justin *Marketplaces and Payments*. URL: http://blog.spreedly.com/2012/11/01/marketplaces-and-payments/#.VJl_ol4gB (acedido a última vez em November 15, 2015)i2012.

[Kim05] Kim Jeeyoung K Factors influencing consumers' apparel purchasing intention in the c2c e-commerce market (2005).

[Kimm13] Kim-Mai Cutler Josh Constine *Facebook Buys Parse To Offer Mobile Development Tools As Its First Paid B2B Service*. URL: http://techcrunch.com/2013/04/25/facebook-parse/ (acedido a última vez em January 12, 2015)i2013.

[King12a] King Brett *Bank 3.0: Why Banking Is No Longer Somewhere You Go But Something You Do* : Hardcover, 2012.

[King12b] King Brett *Mobile payments already exceed cheques*. URL: http://www.finextra.com/blogs/fullblog.aspx?blogid=6560i2012b.

[Knib02] Kniberg Henrik What makes a micropayment solution succeed. In: *Institution for Applied Information Technology. Kista, ...* (2002), pp. 1–68.

[KoPA08] Kousaridas Apostolos , Parissis George and Apostolopoulos Theodore An open financial services architecture based on the use of intelligent mobile devices. In: *Electronic Commerce Research and Applications* vol. 7 (2008), Nr. 2, pp. 232–246.

[KTSK10] Kim Changsu , Tao Wang , Shin Namchul and Kim Ki-Soo An empirical study of customers' perceptions of security and trust in e-payment systems. In: *Electronic Commerce Research and Applications* vol. 9, Elsevier B.V. (2010), Nr. 1, pp. 84–95.

[LiPW06] Linck K , Pousttchi K and Wiedemann DG Security issues in mobile payment from the customer viewpoint. In: *Proceedings of the 14th European Conference on Information Systems (ECIS 2006)* (2006), Nr. 2923.

[LNCL03] Lawrence Elaine , Newton Stephen , Corbitt Brian , Lawrence John , Dann Stephen and Thanasankit Theerasak *Internet commerce : digital models for business*. 3. ed. : John Wiley & Sons, 2003 — ISBN 0470802359 9780470802359.

[Logu13] Logue Peter J. *THE IMPACT OF PCI DSS ON CREDIT CARD FRAUD AND MERCHANTS IN THE U.S.*, Faculty of Utica College, 2013.

[Lore10] LorenzoGatti *Singleton Pattern*. URL: http://c2.com/cgi/wiki?SingletonPatterni2010.

[Marc14] Marcus Wohlsen *San Francisco's New Housing Rules Are the Best Thing to Happen to Airbnb*. URL: http://www.wired.com/2014/10/san-franciscos-new-limits-best-thing-happen-airbnb/ (acedido a última vez em December 21, 2014)i2014.

[Mark01] Markopoulos Dennis Abrazhevich and Abrazhevich Dennis and Denis Abrazhevich and Eindhoven Technische and Dr. P. Electronic Payment Systems: a User-Centered Perspective and Interaction Design. In: *Proceedings of CSCW '02*, 2001, pp. 35–50.

[Nguy14] Nguyen Giang Thu *Exploring collaborative consumption business models-case peer-to-peer digital platforms*, Aalto University, 2014. — ver modelos de negocio .

[Noba14] noBackend *noBackend*. URL: http://nobackend.org/ (acedido a última vez em January 12, 2015)i2014.

[Nune11] Nunes Diogo de Campos *Arquitectura de Aplicações para Projecto de Engenharia em Plataformas Móveis*, Universidade do Porto, 2011.

[Orac14] Oracle *O que é J2ME ou Java ME?*. URL: https://www.java.com/pt_BR/download/faq/whatis_j2me.xml (acedido a última vez em November 11, 2014)i2014.

[Pars15a] Parse *Cloud Code*. URL: https://parse.com/docs/js/guide#cloud-codei2015a.

[Pars15b] Parse *iOS Developer - Security*. URL: https://parse.com/docs/ios/guide#securityi2015b.

[Paym15] Payments Switch *Switch Payments*. URL: https://switchpayments.com/#introi2015.

[Pcis13] PCI Security Standards Council Requisitos e procedimentos da avaliação de segurança (2013).

[Pcis14] PCI Security Standards Council *Sobre o PCI Security Standards Council*. URL: https://pt.pcisecuritystandards.org/minisite/en/about.php (acedido a última vez em December 05, 2014)i2014.

[Pere14] Perez Sarah *With myDoorman, You Never Have To Miss A Package Delivery Again*. URL: http://techcrunch.com/2014/01/08/with-mydoorman-you-never-have-to-miss-a-package-delivery-again/ (acedido a última vez em December 22, 2014)i2014.

[RBZJ02] Rocha RA da , Bortoluzzi AC , Zanini MRK and Júnior NJZ A internet ea reinvenção do mundo dos negócios. In: *XXII Encontro Nacional de Engenharia de Produção* (2002), pp. 1–8.

[Rich12] Richter Felix *Two Thirds of Android Users Don't Pay for Apps*. URL: http://www.statista.com/chart/558/amount-of-money-spend-on-apps-by-android-and-ios-users/ (acedido a última vez em January 10, 2015)i2012.

[Schn52] Schneider Gary P. *Electronic commerce*. 10. ed. : Learning, MA Course Technology Cengage, 1952 — ISBN 9781133526827.

[Séne15] Séneca Hugo Tribunal manda bloquear site da Uber em Portugal. In: *Exame Informática* (2015).

[Stat14] Statista *Paypal: active registered accounts 2010-2014*. URL: http://www.statista.com/statistics/218493/paypals-total-active-registered-accounts-from-2010/ (acedido a última vez em December 18, 2014)i2014.

[Sund12] Sundararajan Arun *Why the Government Doesn't Need to Regulate the Sharing Economy*. URL: http://www.wired.com/2012/10/from-airbnb-to-coursera-why-the-government-shouldnt-regulate-the-sharing-economy/ (acedido a última vez em November 27, 2015)i2012.

[Task14] TaskRabbit *My Tasker didn't show up/is very Late.* URL: https://taskrabbit.zendesk.com/entries/59868274-My-Tasker-didn-t-show-up-is-very-Late- (acedido a última vez em November 26, 2014)i2014.

[Tele00] Telecom Scientific *Características do primeiro telemóvel*. URL: https://sites.google.com/site/scientifictelecom/caracteristicas-do-primeiro-telemovel (acedido a última vez em January 05, 2015).

[TsSt05] Tsiakis Theodosios and Sthephanides George The concept of security and trust in electronic payments. In: *Computers & Security* vol. 24 (2005), Nr. 1, pp. 10–15.

[Ultr15] UL Transaction Securtity Division Tokenization Explained (2015).

[Wrig02] Wright D. Comparative evaluation of electronic payment systems. In: *INFOR* vol. 40 (2002), Nr. 1, pp. 71–85.

Anexo A

Normas PCI DSS

- Construir e manter a segurança de rede e sistema:
  - Instalar e manter uma configuração de *firewall* para proteger os dados do titular do cartão
  - Não usar padrões disponibilizados pelo fornecedor para senhas do sistema e outros parâmetros de segurança
- Proteger os dados do titular do cartão
  - Proteger os dados armazenados do titular do cartão
  - Criptografar a transmissão dos dados do titular do cartão em redes abertas e públicas
- Manter um programa de gestão de vulnerabilidades
  - Usar e atualizar regularmente o *software* ou programas de antivírus
  - Desenvolver e manter sistemas e aplicativos seguros
- Implementar medidas rigorosas de controlo de acesso
  - Restringir o acesso aos dados do titular do cartão de acordo com a necessidade de conhecimento para o negócio
  - Identificar e autenticar o acesso aos componentes do sistema

- Restringir o acesso físico aos dados do titular do cartão

- Monitorizar e testar as redes regularmente
  - Acompanhar e monitorizar todos os acessos com relação aos recursos da rede e aos dados do titular do cartão
  - Testar regularmente os sistemas e processos de segurança
- Manter uma política de segurança de informações
  - Manter uma política que aborde a segurança das informações para todas as equipas.

Anexo B

Lista de *frameworks noBackend*

- *Backendless*
- *Deployd*
- *Firebase*
- *Hoodie*
- *Kinvey*
- *Parse*
- *RemoteStorage*
- *Sockethub*

Anexo C

Documentação *Switch Payments*

C.1 *SDK Switch Payments iOS*

SwitchPayments

Superclass: NSObject
Declared In: SwitchPayments.h

Introduction

SDK for use with Switch Payment API.

Methods

+createCardTokenWithName:number:month:year:cvv:country:withBlock:
+LIVE
+SANDBOX
+setupWithPublicKey:andEnvironment:

createCardTokenWithName:number:month:year:cvv:country:withBlock:

```
+(void) createCardTokenWithName:(NSString*)name number: (NSNumber*)
number

            month:(NSNumber*)month year:(NSNumber*)year cvv:
(NSString*) cvv

            country:(NSString*)country withBlock:(void(^) (SPCard
*card, NSError *error)) block;
```

Parameters

`name`
: is the name of credit card owner

`number`
: is the number of credit card

`month`
: is the month of validity date of credit card

`year`
: is the year of validity date of credit card

`cvv`
: are the 3 caracter present in back of credit card

`country`
: is the country of the credit card

`block`
: is the block to execute. It should have the following argument signature: `^(SPCard *card, NSError *error)`.

Discussion

Create a Card token with credit card data in background with block

LIVE

```
+ (NSString*) LIVE;
```

Return Value

Return a NSString with URL of Production API

Discussion

Live Environment

SANDBOX

```
+ (NSString*) SANDBOX;
```

Return Value

Return a NSString with URL of SandBox API

Discussion

SandBox Environment

setupWithPublicKey:andEnvironment:

```
+ (void) setupWithPublicKey:(NSString*)publicKey

         andEnvironment:(NSString*)environment;
```

Parameters

`publicKey`
is a public key available in Switch Payment Dashboard

`environment`
is the environment to be used - SANDBOX or LIVE

Discussion

Configure the Switch Payment with key public and the environment to be used

SPCard.h

Includes: <Foundation/Foundation.h>

Introduction

Use the links in the table of contents to the left to access the documentation.

Functions

`property`
Last four numbers of credit card number.

`property(nonatomic, strong)`
Name of credit card owner.

`property(nonatomic, strong)`
Month of validity date of credit card.

`property(nonatomic, strong)`
Year of validity date of credit card.

`property(nonatomic, strong)`
Token create by Switch Payment API.

property

Last four numbers of credit card number.

```
@property (

    nonatomic,

    strong) NSString *last4Numbers;
```

property(nonatomic, strong)

Name of credit card owner.

```
@property (
    nonatomic,
    strong) NSString *name;
```

property(nonatomic, strong)

Month of validity date of credit card.

```
@property (
    nonatomic,
    strong) NSNumber *month;
```

property(nonatomic, strong)

Year of validity date of credit card.

```
@property (
    nonatomic,
    strong) NSNumber *year;
```

property(nonatomic, strong)

Token create by Switch Payment API.

```
@property (
    nonatomic,
    strong) NSString *token;
```

C.2 *Switch Payments Cloud Module Parse*

SP

SwitchPayment Module

SP.initialize(merchant_key, key, environmentType)

Initialize the Switch Payment module with the merchant_key, private_key and environment.

Parameters

merchant_key: `String`, Your merchant key in Switch

key: `String`, Your private key in Switch

environmentType: `String`, Environment to be use. SANDBOX or LIVE

SP.createCard(oneTimeCardToken)

Create a card.

Parameters

oneTimeCardToken: `String`, A one time card token

Returns: `Parse.Promise`

SP.createPayment(amount, currency, card)

Create a Payment capture on creation

Parameters

amount: `double`, Amount of payment

currency: `String`, Currency type

card: `object`, Card create with createCard

Returns: `Parse.Promise`

Anexo D

Enunciado do Teste de Usabilidade

D.1 Aplicação Cliente

1. Registar na aplicação com os seguintes dados:
    - Primeiro nome: Teste
    - Segundo nome: Usabilidade
    - *E-mail*: testeX@cliente.com (x = número do teste)
    - Número de telefone: 919191911
    - *Password*: 123
    - Confirmação de *Password*: 123
2. Terminar sessão
3. Iniciar sessão com:
    - *E-mail*: testeX@cliente.com (x = número do teste)
    - *Password*: 123
4. Escolher o restaurante "Mauritânia" e adicionar o produto "Francesinha" ao carrinho
5. Alterar no carrinho número de unidades do produto "Francesinha" para 2.
6. Realizar *checkout* da encomenda
7. Escolher a localização do mapa referente à localização da empresa *Glazed Solutions*
8. Confirmar encomenda

D.2 Aplicação Estafeta

1. Iniciar sessão com:
    a. *E-mail*: estafeta1@teste.com

   b. *Password*: 123
2. Ver as encomendas já realizadas
3. Classificar um cliente
4. Ver detalhes de uma encomenda
5. Inicial modo de espera
6. Deligar modo de espera
7. Terminar sessão

Anexo E

Resultado dos testes de usabilidade

E.1 Resultados conjuntos

E.1.1 Distribuição da frequência

Tabela E.1: Distribuição da frequência

Intervalo	*Percentagem (%)*
<=5	65,93
[6 – 10]	22,22
[11- 15]	6,67
[16 – 20]	0,00
>20	5,18

E.1.2 Outros valores estatísticos

Tabela E.2: Valores estatísticos

	Resultado (s)
Média	6,10
Máximo	48
Mínimo	1
Desvio Padrão	7,25

E.2 Resultados dos testes à aplicação em modo cliente

E.2.1 Distribuição da frequência

Tabela E.3: Distribuição da frequência dos testes à aplicação em modo cliente

Intervalo	***Percentagem (%)***
<=5	45,84
[6 – 10]	34,72
[11- 15]	11,11
[16 – 20]	0,00
>20	8,33

E.2.2 Outros valores estatísticos

Tabela E.4: Valores estatísticos dos testes à aplicação em modo cliente

	Resultado (s)
Média	8
Máximo	40
Mínimo	1
Desvio Padrão	7,62

E.2.3 Resultados individuais de cada tarefa

Distribuição da frequência

Tabela E.5: Distribuição da frequência por tarefa em modo cliente

Intervalo	***Tarefa***							
	1	***2***	***3***	***4***	***5***	***6***	***7***	***8***
<=5	22,22	66,67	66,67	11,11	22,22	77,78	11,11	88,89
[6 – 10]	44,44	22,22	33,33	44,44	66,67	11,11	44,44	11,11
[11- 15]	11,11	0,00	0,00	33,33	11,11	0,00	33,33	0,00
[16 – 20]	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
>20	22,22	11,11	0,00	11,11	0,00	11,11	11,11	0,00

Outros valores estatísticos

Tabela E.6: Valores estatísticos por tarefa em modo cliente

	Tarefa							
	1	***2***	***3***	***4***	***5***	***6***	***7***	***8***
Média	11,56	7,22	3,78	12,56	8	7	11,89	2
Máximo	28	26	10	31	13	40	25	6
Mínimo	2	1	1	5	5	1	5	1
Desvio Padrão	8	7	3,01	7,24	2,45	12,01	5,67	1,63

E.3 Resultados dos testes à aplicação em modo estafeta

E.3.1 Distribuição da frequência

Tabela E.7: Distribuição da frequência dos testes à aplicação em modo estafeta

Intervalo	***Percentagem (%)***
<=5	88,89
[6 – 10]	7,94
[11- 15]	1,59
[16 – 20]	0,00
>20	1,59

E.3.2 Outros valores estatísticos

Tabela E.8: Valores estatísticos dos testes à aplicação em modo estafeta

	Resultado (s)
Média	3,92
Máximo	48,00
Mínimo	1,00
Desvio Padrão	6,10

E.3.3 Resultados individuais de cada tarefa

Distribuição da frequência

Tabela E.9: Distribuição da frequência por tarefa em modo estafeta

Intervalo	***Tarefas***						
	1	***2***	***3***	***4***	***5***	***6***	***7***
<=5	100,00	88,89	44,44	100,00	88,89	100,00	100,00
[6 – 10]	0,00	11,11	33,33	0,00	11,11	0,00	0,00
[11- 15]	0,00	0,00	11,11	0,00	0,00	0,00	0,00
[16 – 20]	0,00	0,00	0,00	0,00	0,00	0,00	0,00
>20	100,00	88,89	44,44	100,00	88,89	100,00	100,00

Outros valores estatísticos

Tabela E.10: Valores estatísticos por tarefa em modo estafeta

	Tarefas						
	1	***2***	***3***	***4***	***5***	***6***	***7***
Média	3,44	3,44	11,00	3,33	3,11	1,67	1,44
Máximo	5,00	8,00	48,00	5,00	6,00	4,00	4,00
Mínimo	1,00	1,00	1,00	1,00	1,00	1,00	1,00
Desvio Padrão	1,83	2,06	13,56	1,41	1,45	1,05	0,96